TECHNISCHE
UNIVERSITÄT
DARMSTADT

Lebenszyklusinformationen von Wissensdokumenten

Erfassung, Verwaltung und Validierung

Vom Fachbereich 18
Elektrotechnik und Informationstechnik
der Technischen Universität Darmstadt
zur Erlangung des Grades eines
Doktor-Ingenieurs (Dr.-Ing.)

genehmigte Dissertationsschrift
von

Dipl.-Ing. Lasse Lehmann

geboren am 28. Mai 1979 in VS-Villingen

Vorsitz: Prof. Dr.-Ing. Roland Werthschützky
Erstreferent: Prof. Dr.-Ing. Ralf Steinmetz
Korreferent: Prof. Dr.-Ing. Matthias Hemmje

Tag der Einreichung: 22.01.2010
Tag der Disputation: 26.03.2010

Darmstadt 2010
Hochschulkennziffer D17

Zusammenfassung

Mit der wachsenden Zahl digital verfügbarer Dokumente wachsen auch die Probleme der Nutzer, die Dokumente persönlich oder in der Gruppe zu organisieren. Insbesondere für Wissensarbeiter ist jedoch ein schnelles Auffinden von für ihre Arbeit relevanten Dokumenten wichtig, um effektiv arbeiten zu können. Nutzer haben aber in vielen Fällen Probleme, Dokumente, die sie oder Gruppenmitglieder gespeichert haben, wiederzufinden. Dies führt sogar so weit, dass Nutzer Dokumente, die sie aus dem Internet heruntergeladen und im Dateisystem gespeichert haben, lieber erneut im Internet suchen, als auf dem lokalen Rechner. Oft wissen sie auch nicht, dass Dokumente, in denen Gruppenmitglieder ihr Wissen dokumentiert haben, überhaupt existieren.

Ein Grund für die schlechte Auffindbarkeit von lokal verwalteten Dokumenten ist, dass nur wenige zusätzliche Informationen über solche Wissensdokumente verfügbar sind. Die Metadaten der Dokumente werden kaum gepflegt und enthalten in den meisten Fällen nicht mehr als die vom Betriebssystem oder einer Applikation zur Bearbeitung des jeweiligen Dokumenttyps automatisiert erzeugten Informationen. Diese sind zumeist wenig aussagekräftig, so dass sie für eine Verbesserung der oben genannten Situation oft nicht geeignet sind. Zudem werden Nutzer durch für die Suche und Organisation der Dokumente verwendete Werkzeuge, wie zum Beispiel den Windows Dateisystem-Explorer, nicht ausreichend unterstützt.

Diese Arbeit basiert auf der Beobachtung, dass eine Vielzahl von Informationen über Dokumente durch Aktionen entstehen, die auf einem Dokument durchgeführt werden. So wird ein Dokument beispielsweise geöffnet, gelesen, bearbeitet oder genutzt. Während dieser Prozesse entstehen Informationen, die für die Verwaltung oder zur Unterstützung des Auffindens der Dokumente nutzbar sind. Meist ist es so, dass die Informationen verloren gehen, wenn sie nicht während der entsprechenden Prozesse erfasst und gespeichert werden. Eine manuelle Erfassung der Informationen findet aufgrund des hohen Aufwands nicht statt. Deshalb verfolgt die vorliegende Dissertation den Ansatz, automatisiert Metadaten aus Prozessen zu gewinnen, die während seines Lebenszyklus auf einem Wissensdokument ablaufen, und die so gewonnenen Informationen entsprechend zu verwalten und nutzbar zu machen.

Hierzu wird zunächst analysiert, welche Informationen während des Lebenszyklus eines Wissensdokuments entstehen. Es wird auf Basis bestehender Lebenszyklusmodelle ein Lebenszyklusmodell für Wissensdokumente entwickelt. Lebenszyklusinformationen werden definiert und in Verwendungs- und Beziehungsinformationen unterteilt. Anhand des Lebenszyklusmodells werden Informationen, die in den verschiedenen Phasen entstehen, identifiziert. Das Hauptaugenmerk liegt in der vorliegenden Dissertation auf Beziehungsinformationen, die bei der Wiederverwendung von Wissensdokumenten entstehen.

Bevor Lebenszyklusinformationen genutzt werden können, müssen sie erfasst, entsprechend verwaltet und systemübergreifend zugänglich gemacht werden. Schließlich ist sicherzustellen, dass die erfassten Informationen ihre Gültigkeit behalten. Alle diese Aspekte werden in der vorliegenden Arbeit berücksichtigt. Es wird ein Framework für die automatisierte Erfassung, die Verwaltung und Nutzung von Lebenszyklusinformationen konzipiert, umgesetzt und evaluiert. Dieses Framework beinhaltet ein auf Plug-ins

basierendes Konzept zur Erfassung der Informationen, welches auf fast beliebige Applikationen übertragbar ist. Zwei verschiedene Konzepte für die Erfassung von Informationen werden identifiziert und in Form von Erfassungskomponenten für drei verschiedene Applikationen umgesetzt. Die Verwaltung und Bereitstellung der erfassten Informationen erfolgt dabei serverbasiert. Für die Verwaltung der Informationen wird in der Arbeit ein Schema zur Verwaltung von Lebenszyklusinformationen vorgestellt, das insbesondere die Erfassung und Verwaltung von Beziehungsinformationen abdeckt, wofür bisher noch keine adäquate Lösung existiert. Darüber hinaus werden Konzepte für verschiedene Nutzungsszenarien von Lebenszyklusinformationen entwickelt und prototypisch für zwei dieser Szenarien umgesetzt.

Gerade im Fall von Beziehungsinformationen ist es notwendig, die Gültigkeit der erfassten Informationen zu gewährleisten. Wenn durch eine Aktion eine Beziehung zwischen zwei Dokumenten entstehen kann, so kann es auch eine Aktion geben, durch welche diese Beziehung ihre Gültigkeit verliert. Um dies zu adressieren, werden in dieser Arbeit zwei Validierungsalgorithmen für Beziehungsinformationen vorgestellt und auf unterschiedlichen Korpora evaluiert. Dabei wird gezeigt, dass die entworfenen Algorithmen auf den getesteten Korpora bessere Ergebnisse liefern als State-of-the-Art-Ansätze. Es wird zudem gezeigt, dass die entworfenen Algorithmen in verschiedenen weiteren Anwendungsszenarien nutzbar sind.

Die im Rahmen der Arbeit durchgeführte nutzerbasierte Evaluation des umgesetzten Frameworks zeigt, dass eine Erfassung valider Lebenszyklusinformationen mit hoher Verlässlichkeit durchführbar ist. Die vorliegende Arbeit schafft also durch die automatische Erfassung von Lebenszyklusinformationen von Wissensdokumenten die Voraussetzung und Grundlage für eine Nutzung dieser zusätzlichen Informationen in vielen Szenarien.

Abstract

With the growing number of documents which are digitally available, users tend to have more and more problems to organize these documents personally or collaboratively. Specifically for knowledge workers, fast retrieval of relevant documents is essential. In many cases, users have problems to rediscover documents that they or members of their working group once stored somewhere. Often, users rather download documents from the Internet once again instead of searching them in their local file system. Often they do not even know that documents where members of their working group documented relevant knowledge even exist.

One reason for the bad retrievability of documents organized in file systems is the lack of information about such documents. Metadata is hardly maintained and usually consists of information automatically provided by the operating system only. This information is barely distinctive and does not help to improve the situation described above. Furthermore, existing tools for retrieval and management of documents like the Windows Filesystem Explorer do not support users sufficiently.

The present thesis is based on the observation that a multitude of information emerges from actions performed on a document during its lifecycle. Users open, read, edit or use documents several times. These processes provide for the emergence of information that can be utilized to support both the management and the retrieval of these documents. However, most of the information is lost if it is not captured during those processes. Manual creation of such information would be too much effort and too costly. The underlying approach of this thesis is the automatic capture of information emerging from processes conducted on a document during its lifecycle. The thus acquired information should then be organized, processed and made accessible for utilization.

Initially, we analyze which kinds of information emerge during the lifecycle of a knowledge document. Based on existing lifecycle models from other domains we develop a lifecycle model for knowledge documents. We define the concept of lifecycle information and categorize it further into relation and usage information. On this basis we identify the information that emerges during the different phases of a document's lifecycle. Hereby, we focus on relation information emerging from reuse processes conducted on knowledge documents.

Before lifecycle information can be utilized, it has to be captured, managed and made accessible across different systems and applications. Finally, we have to make sure that the relations captured stay valid. All of these aspects are addressed in the given thesis. We have designed, implemented and evaluated a framework for automatic capture, management and utilization of lifecycle information. The framework deploys a plug-in-based concept for the capture of information that is portable to arbitrary applications. We have identified two different means to capture valid relations and implemented both in three different capture components. The management and provision of captured information is done in a server-based manner. We propose a scheme for the organization of lifecycle information, which specifically covers the capture and management of relation information, for which no sufficient solution

existed so far. Furthermore, we have designed various scenarios for the utilization of lifecycle information and implemented two of them prototypically.

Especially for relation information it is necessary to ensure the validity of the information captured. If there is an action that provides for the emergence of a relation there might also be actions that cause the relation to become invalid. To address this issue we have designed two algorithms for the automatic validation of relation information and have evaluated them on different corpora. On the given corpora our algorithms perform better quality-wise than state of the art approaches while maintaining a lower storage consumption. We furthermore show that the proposed algorithms can be applied in various additional scenarios.

The user-based evaluation of the proposed framework we have conducted shows that the capture of valid lifecycle information is achievable with high reliability. Through the automatic capture of lifecycle information of knowledge documents, this thesis creates a basis and prerequisite for the utilization of this new kind of information in various scenarios.

Danksagung

An erster Stelle möchte ich mich bei meinem Doktorvater Ralf Steinmetz für die Möglichkeit der Promotion und das ideale und inspirierende Umfeld bedanken sowie bei meinem Korreferenten Matthias Hemmje für seine wertvollen und umfangreichen Hinweise.

Meinem Gruppenleiter Christoph Rensing danke ich für die großartige Unterstützung und Anleitung während meiner gesamten Zeit bei KOM. Vielen Dank auch an Tomas, der mit seinen Anregungen einen großen Einluß auf meine Arbeit hatte. Meiner langjährigen Zimmergenossin Sonja möchte ich auf diesem Weg für die schöne gemeinsame Zeit, viele angeregte Diskussionen und dafür dass sie mich so lange ausgehalten hat, danken. Großer Dank gebührt weiterhin allen weiteren aktuellen und ehemaligen Mitgliedern der Knowledge-Media-Gruppe bei KOM, besonders Stephan, Philipp und Doreen, die während meiner Zeit bei KOM mehr als nur Kollegen für mich waren.

Natürlich danke ich ebenso allen anderen Kolleginnen und Kollegen bei KOM für die schöne Zeit und die tolle Atmosphäre. Viele von ihnen haben mich durch Gespräche, Diskussionen oder sonstige Aktivitäten unterstützt, oder waren mir anderweitig hilfreich. Andreas Faatz und Stefan Hoermann, als ehemalige Mitglieder des Lehrstuhls, möchte ich dafür danken, dass sie mich zu KOM gebracht und mich davon überzeugt haben, den Weg der Promotion einzuschlagen.

Vielen Dank auch allen, die die vorliegende Arbeit korrekturgelesen haben, insbesondere meiner Schwester Mailin und Wilfried Heupel. Besonderer Dank gilt meinen Eltern, Geschwistern und Freunden, die mich immer unterstützt haben. Aus tiefstem Herzen danke ich meiner Frau Siri und meinen Kindern Luna und Anuk für die emotionale Unterstützung und die immer willkommenen Gründe zur Ablenkung. Schließlich möchte ich an dieser Stelle noch all meinen Studenten, insbesondere Arno, für die gute Zusammenarbeit danken.

Inhaltsverzeichnis

Teil I.

Der Lebenszyklus und Lebenszyklusinformationen von Wissensdokumenten

1 Einleitung

1.1 Motivation

Mit der wachsenden Zahl digital verfügbarer Dokumente wachsen auch die Probleme der Nutzer, die Dokumente persönlich oder in der Gruppe zu organisieren. Insbesondere für Wissensarbeiter ist jedoch ein schnelles Auffinden von für ihre Arbeit relevanten Dokumenten wichtig, um effektiv arbeiten zu können. Nutzer haben aber in vielen Fällen Probleme, Dokumente, die sie oder Gruppenmitglieder gespeichert haben, wiederzufinden. Dies führt sogar so weit, dass Nutzer Dokumente, die sie aus dem Internet heruntergeladen und im Dateisystem gespeichert haben, lieber erneut im Internet suchen, als auf dem lokalen Rechner [NP05]. Oft wissen sie auch nicht, dass Dokumente, in denen Gruppenmitglieder ihr Wissen dokumentiert haben, überhaupt existieren.

Ein Grund für die schlechte Auffindbarkeit von lokal verwalteten Dokumenten ist, dass nur wenige zusätzliche Informationen über solche Wissensdokumente verfügbar sind. Die Metadaten der Dokumente werden kaum gepflegt und enthalten in den meisten Fällen nicht mehr als die vom Betriebssystem oder einer Applikation zur Bearbeitung des jeweiligen Dokumenttyps automatisiert erzeugten Informationen. Diese sind zumeist wenig aussagekräftig, so dass sie für eine Verbesserung der oben genannten Situation oft nicht geeignet sind. Zudem werden Nutzer durch für die Suche und Organisation der Dokumente verwendete Werkzeuge, wie zum Beispiel den Windows Dateisystem-Explorer, nicht ausreichend unterstützt. Desktop-Suchmaschinen, wie *Google Desktop* oder *Windows Desktop Search*, versuchen, die Auffindbarkeit der Dokumente mit Hilfe eines Volltextindex der Dokumenteninhalte zu verbessern. Dieser hilft jedoch überwiegend bei textbasierten Dokumenten. Außerdem erzeugt die Indexierung aller Inhalte oftmals Rauschen, wodurch die Qualität der Ergebnisse wiederum gemindert werden kann [BM90].

Ein weiterer Grund, weshalb die Suche im Internet so viel effektiver als zum Beispiel die Suche in Dateisystemen funktioniert, ist ein anderer: Dokumente im Internet sind miteinander verknüpft - die Beziehungen der Dokumente untereinander sind gegeben durch die Hypertext-Struktur des World Wide Web und werden durch diese expliziert. Diese Beziehungen werden von Suchmaschinen, angefangen mit der erfolgreichsten Suchmaschine Google, zusätzlich zum Volltextindex genutzt, um die Ergebnisse, die die Suchmaschine zurückliefert, entsprechend zu priorisieren. Hierfür wird im Fall von Google der PageRank-Algorithmus genutzt [PBMW99]. Solche Beziehungen zwischen Wissensdokumenten auf lokalen PCs sind jedoch im Gegensatz zum World Wide Web nicht explizit vorhanden und können folglich nicht genutzt werden.

Um das genannte Problem der fehlenden Informationen zu adressieren, ist es notwendig, zusätzliche Informationen über solche Dokumente zu gewinnen. Es gibt diverse Ansätze, die versuchen, solche Metadaten zu Dokumenten im Nachhinein zu gewinnen (zum Beispiel [MOD07, MZ07, LAH+02, Alb08, Ste02]). Diese Arbeit basiert auf der Beobachtung, dass eine Vielzahl von Informationen durch Aktionen entstehen, die auf einem Dokument durchgeführt werden. So wird ein Dokument beispielsweise geöffnet, gelesen, bearbeitet oder genutzt. Während dieser Prozesse entstehen Informationen, die für die

Verwaltung oder zur Unterstützung des Auffindens der Dokumente nutzbar sind. Meist ist es so, dass die Informationen verloren gehen, wenn sie nicht während der entsprechenden Prozesse erfasst und gespeichert werden. Eine manuelle Erfassung der Informationen findet aufgrund des hohen Aufwands nicht statt. Deshalb verfolgt diese Arbeit den Ansatz, (1) automatisiert Metadaten aus Prozessen zu gewinnen, die ohnehin während seines Lebenszyklus auf einem Wissensdokument ablaufen, und (2) die so gewonnenen Informationen entsprechend zu verwalten und nutzbar zu machen.

Ein wichtiger Typus der genannten Prozesse ist die Wiederverwendung. Viele Wissensdokumente werden nicht komplett neu erstellt. Einige entstehen durch Überarbeitung eines bestehenden Dokuments, in vielen Fällen aber werden zumindest Teile aus bestehenden Dokumenten in ein neues Dokument eingefügt. Oftmals ist sich der ursprüngliche Autor des Dokuments der Wiederverwendung nicht bewusst. Ein solcher Wiederverwendungsprozess impliziert grundsätzlich eine Beziehung zwischen dem Quelldokument und dem Zieldokument des Prozesses. Durch die Erfassung von Informationen, die aus diesen Wiederverwendungsprozessen entstehen, können Beziehungen zwischen Dokumenten expliziert werden. Diese Beziehungen können auf unterschiedliche Weise genutzt werden. Dazu zählt die Verbesserung der Suche in Dateisystemen, beispielsweise durch entsprechende Ranking-Verfahren, ähnlich dem PageRank-Verfahren bei der Internet-Suche, oder durch Anreicherung von Suchergebnissen oder Generierung von Vorschlägen mittels der gewonnenen Beziehungen.

1.2 Ziele der Arbeit

Wenn Lebenszyklusinformationen eines Dokuments nicht erfasst werden, gehen sie in der Regel verloren. Daher ist es das vordergründige Ziel dieser Arbeit, die Erfassung von Lebenszyklusinformationen für Wissensdokumente automatisiert und ohne Eingreifen des Nutzers zu ermöglichen. Dies soll insbesondere im Hinblick auf Wiederverwendung, und die dabei entstehenden Beziehungen zwischen Dokumenten geschehen. Weiterhin ist es notwendig, die Informationen so zu verwalten, dass sie systemübergreifend, z.B. in einer Gruppe zugreifbar sind. Schließlich müssen die Informationen entsprechend verarbeitet und aufbereitet werden, um genutzt werden zu können. Die erfassten Informationen sollen schließlich zur Nutzung zur Verfügung gestellt werden, um Verwaltung, Auffinden oder Organisation von Wissensdokumenten zu vereinfachen.

Das Hauptziel der Arbeit ist es also, durch die automatische Erfassung von Lebenszyklusinformationen von Dokumenten eine Grundlage für diverse Nutzungsszenarien zu schaffen, um den Umgang mit diesen Dokumenten zu erleichtern.

Szenarien für die Nutzung zu entwerfen und umzusetzen und anhand dieser die Nutzbarkeit der erfassten Informationen zu evaluieren, ist hingegen nicht Ziel dieser Arbeit. Des Weiteren liegt die Behandlung von Datenschutz- und Datensicherheitsaspekten ebenso wenig im Fokus dieser Dissertation.

1.3 Beiträge

Die vorliegende Arbeit schafft durch die automatische Erfassung von Lebenszyklusinformationen von Wissensdokumenten die Voraussetzung und Grundlage für eine Nutzung dieser zusätzlichen Informationen in vielen Szenarien. Hierzu tragen folgende Aspekte der Arbeit bei:

- Um Lebenszyklusinformationen automatisiert erfassen zu können, müssen sie zunächst identifiziert werden, d.h. es muss analysiert werden, während welcher Prozesse, in welchen Applikationen, zu welchen Zeitpunkten welche Informationen entstehen. Hierzu wurde ein *Lebenszyklusmodell* für Wissensdokumente entwickelt, auf dessen Basis Lebenszyklusinformationen definiert werden konnten. Zusätzlich wurde eine Einteilung und Kategorisierung vorgenommen.
- Um Lebenszyklusinformationen für eine Nutzung zur Verfügung stellen zu können, ist ein Format notwendig, mit dessen Hilfe die Informationen organisiert und verwaltet werden können. Es wurde ein entsprechendes *Schema* zur Verwaltung von Lebenszyklusinformationen entwickelt, das kompatibel zum verbreiteten CAM-Schema [CAM07] ist, und insbesondere die Erfassung und Verwaltung von Beziehungsinformationen abdeckt.
- Gerade im Fall von Beziehungsinformationen ist es notwendig, die Gültigkeit der erfassten Informationen zu gewährleisten. Wenn durch eine Aktion eine Beziehung zwischen zwei Dokumenten entstehen kann, so kann es auch eine Aktion geben, durch welche diese Beziehung ihre Gültigkeit verliert. Um dies zu adressieren, wurden zwei Validierungsalgorithmen für Beziehungsinformationen entworfen, umgesetzt und auf unterschiedlichen Korpora evaluiert. Die entworfenen Algorithmen können zusätzlich noch in diversen weiteren Anwendungsszenarien eingesetzt werden und bieten im Bereich der Erkennung von Wiederverwendung eine Verbesserung gegenüber State-of-the-Art-Algorithmen.
- Es wurde ein Framework für die Erfassung, Verwaltung und Nutzung von Lebenszyklusinformationen konzipiert, umgesetzt und evaluiert. Das Framework beinhaltet ein Plug-in basiertes Konzept zur Erfassung der Informationen, welches auf fast beliebige Applikationen übertragbar ist.
- Als Proof-of-Concept wurden Komponenten für die Erfassung von Lebenszyklusinformationen für drei verschiedene Applikationen und eine robuste serverbasierte Lösung für die Verwaltung der Informationen umgesetzt.
- Schließlich wurde das umgesetzte System hinsichtlich der Erfassung von Beziehungsinformationen in einer Benutzerstudie evaluiert.

1.4 Struktur der Arbeit

Die vorliegende Arbeit gliedert sich in vier Hauptteile. Nach der Einleitung erfolgen in Kapitel 2 die Analyse und die Definition des Gegenstands der Arbeit: Lebenszyklusinformationen von Wissensdokumenten. Hierzu werden zunächst Wissensdokumente definiert und von anderen Dokumenten abgegrenzt. Aus verschiedenen existierenden Modellen wird ein geeignetes Lebenszyklusmodell für Wissensdokumente entwickelt. Anhand dieses Modells werden Lebenszyklusinformationen definiert, kategorisiert und analysiert. Auf Basis der Ergebnisse des ersten Teils der Arbeit wird im zweiten Teil ein Framework zur Erfassung, Verwaltung und Nutzung von Lebenszyklusinformationen konzipiert (Kapitel 4). Zunächst erfolgt in Kapitel 3 jedoch eine Analyse bestehender Systeme und verwandter Arbeiten, die sich mit der Erfassung und Nutzung von Informationen aus dem Lebenszyklus von Dokumenten befassen.

Im dritten Teil der Arbeit geht es um die Validierung von Beziehungsinformationen als Teilmenge der Lebenszyklusinformationen. Die vorgestellten Techniken gewährleisten die Gültigkeit erfasster Beziehungsinformationen. In Kapitel 5 wird ein neuer, auf bestehenden Ansätzen aufbauender String-Matching-Algorithmus vorgestellt und evaluiert. Kapitel 6 beschreibt eine neue Fingerprinting-Methode, die neben der Nutzung zur Gültigkeitsprüfung von Beziehungsinformationen in diversen weitere Anwendungsfällen eingesetzt werden kann.

Der vierte und abschließende Teil der Arbeit beschreibt die Umsetzung des in Teil II der Arbeit konzipierten Frameworks zur Erfassung, Verwaltung und Nutzung von Lebenszyklusinformationen (Kapitel 7) sowie dessen nutzerbasierte Evaluation in Kapitel 8. Kapitel 9 schließt die Arbeit mit einer Zusammenfassung und einem Ausblick auf zukünftige Arbeiten ab.

2 Lebenszyklusinformationen

In diesem Kapitel erfolgt zunächst eine tiefergehende Motivation des gewählten Ansatzes. Hierfür wird das zu Grunde liegende Anwendungsszenario anhand eines Beispiels verdeutlicht. Im darauf folgenden Abschnitt erfolgt die Definition und Erläuterung grundlegender Begrifflichkeiten wie "Wissensdokument" oder "Wiederverwendung". Aufbauend auf bestehenden Modellen des Lebenszyklus für Dokumente oder Lernobjekte wird ein Lebenszyklusmodell für Wissensdokumente entwickelt, das dem Zweck dient, die innerhalb des Lebenszyklus eines Wissensdokuments entstehenden Informationen identifizierbar und formalisierbar zu machen. Dies geschieht im letzten Teil des Kapitels. Davor erfolgt die Definition des Begriffs "Lebenszyklusinformationen" und dessen Positionierung zum Begriff "Metadaten". Zum Abschluss des Kapitels werden auf Basis der vorgenommenen Analyse Kriterien für ein System zur Erfassung und Nutzbarmachung von Lebenszyklusinformationen formuliert.

2.1 Beispielszenario und Motivation

Im Folgenden wird beispielhaft der Lebenszyklus einer Folienpräsentation dargestellt. Anhand dieses Beispiels kann verdeutlicht werden, welche Arten von Informationen zu welchen Zeitpunkten im Lebenszyklus entstehen. Der Lebenszyklus der Beispielpräsentation ist in Abbildung 2.1 dargestellt. Jeder der abgebildeten Schritte wird im Folgenden erläutert und ist entsprechend gekennzeichnet.

Ein Wissenschaftler soll einen Vortrag für eine Konferenz vorbereiten. Hierbei geht er wie folgt vor: Er legt eine neue Präsentation an und erstellt zunächst eine grobe inhaltliche Struktur des Vortrags (1). Da er bereits Vorträge zu verschiedenen anderen Aspekten seiner Arbeit vorbereitet hat, durchsucht er zunächst seinen lokalen Rechner nach diesen Vorträgen und sucht Folien heraus, die er für den geplanten Vortrag wiederverwenden kann. Außerdem hat er im Rahmen seiner Arbeit Messungen durchgeführt. Einige Diagramme seiner Messungen hat er bereits publiziert. Er sucht das betreffende Dokument und kopiert die gewünschten Diagramme in die Präsentation. Weitere Ergebnisse bezieht er aus der Tabellenkalkulation, die er nutzt, um seine Messergebnisse zu verwalten (2). Um seine Arbeit in den Kontext seiner Forschungsgruppe zu stellen, verwendet er zudem Folien aus Präsentationen seiner Kollegen. So kann er die inhaltlichen Zusammenhänge der verschiedenen Arbeiten innerhalb der Gruppe besser darstellen. Diese findet er im zentralen Dateisystem seiner Forschungsgruppe (3). Da er in der Präsentation seine Arbeit in den Zusammenhang verwandter Arbeiten anderer Forscher stellen will, sucht er auch hierzu nach Veröffentlichungen und Präsentationen und findet sie auf den Seiten der Forscher im Internet. Einzelne, besonders repräsentative Teile daraus, z.B. Abbildungen oder Zitate, verwendet er ebenso für seine Präsentation (4). Der Autor überarbeitet die wiederverwendeten Inhalte und fügt Text und Grafiken hinzu. Folien, zu denen noch keine Inhalte existieren, erstellt er komplett neu (5). Der gesamte Vorgang geschieht in mehreren Iterationen, an verschiedenen Tagen. Dabei speichert der Autor die Präsentation immer unter dem jeweiligen Datum, so dass an jedem Tag, an dem er an den Folien arbeitet, eine neue Version der Präsentation entsteht. Der Autor schickt die Präsentation an seinen Betreuer oder Kol-

legen, um Verbesserungsvorschläge zu erhalten und stellt die Präsentation unter Berücksichtigung dieser Vorschläge fertig (6). Schließlich hält der Wissenschaftler den Vortrag im Rahmen der Konferenz. In derselben Session werden weitere Vorträge anderer Wissenschaftler zu verwandten Themen gehalten (7). Nach der Konferenz legt der Wissenschaftler die Präsentation im zentralen Filesystem der Forschungsgruppe ab (8). Während des gesamten Zeitraums, von der Erstellung bis zur Nutzung der Präsentation

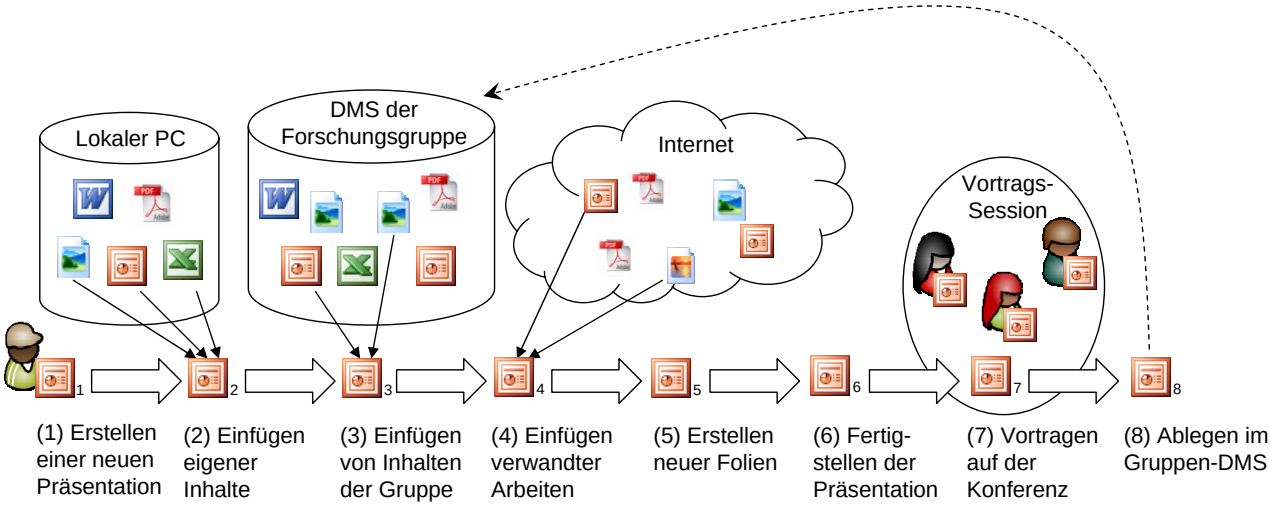

Abbildung 2.1.: Beispiellebenszyklus einer PowerPoint-Präsentation

und deren Archivierung, also während ihres gesamten Lebenszyklus, entstehen für eine spätere Nutzung potentiell interessante Informationen. So wird die Präsentation zum Beispiel in einer bestimmten Applikation (z.B. Microsoft PowerPoint), auf einem bestimmten Rechner erstellt. Dabei entstehen Informationen über die Dauer der Bearbeitung, die dazu verwendeten Applikationen sowie das System, auf dem dies geschieht. Das Dokument wird gegebenenfalls von verschiedenen Personen, auf unterschiedlichen Systemen bearbeitet oder genutzt. Durch die Nutzung selbst, also das Vortragen der Präsentation, fallen weitere Informationen an, beispielsweise die Dauer der Präsentation oder das Datum.

Wie die Präsentation im Beispiel, werden viele Wissensdokumente nicht von Grund auf neu erstellt, sondern bereits bestehende Dokumente als Grundlage verwendet oder Teile aus bereits bestehenden Dokumenten kopiert und in das neue Dokument eingefügt. Dadurch entstehen Beziehungen zwischen dem neuen Dokument und verschiedensten weiteren Dokumenten auf unterschiedlichen Systemen. In dem gegebenen Beispiel entstehen Beziehungen zu eigenen Dokumenten auf dem lokalen Rechner des Wissenschaftlers, zu Dokumenten der Kollegen aus der Forschungsgruppe im zentralen Dateisystem sowie zu den Dokumenten der Forscher, die sich mit verwandten Themen beschäftigen, im Internet. Wiederverwendung ist also ein wichtiger Aspekt im Lebenszyklus eines Wissensdokuments und impliziert Beziehungen zwischen den beteiligten Dokumenten.

Die meisten der genannten Informationen werden jedoch derzeit mit den für die Nutzung und Bearbeitung der Dokumente verwendeten Applikationen nicht erfasst. Viele solcher Informationen sind allerdings nicht im Nachhinein rekonstruierbar. So ist beispielsweise die Dauer eines Vortrags, das System, auf dem ein Dokument bearbeitet wurde oder der Ursprung der wiederverwendeten Diagramme, anhand des zugehörigen Präsentationsdokuments nicht wieder herstellbar.

Viele dieser Informationen können sinnvoll für die Verwaltung oder das Retrieval von Wissensdokumenten genutzt werden. Dieses Kapitel widmet sich daher der Definition und Formalisierung dieser Lebenszyklusinformationen. Dies geschieht auf Basis eines Lebenszyklusmodells für Wissensdokumente.

2.2 Grundlegende Definitionen

Dieser Abschnitt widmet sich der Definition der grundlegenden Begrifflichkeiten. Es erfolgt zunächst die Definition des Begriffs "Wissensdokument" sowie dessen Einordnung und Abgrenzung zu den Definitionen der verwandten Begriffe "Lernobjekt" und "Lernressource". Darauf folgend werden Autorenprozesse aus dem E-Learning-Bereich, soweit sie zum Verständnis dieser Arbeit notwendig sind erläutert und für Wissensdokumente verallgemeinert. Schließlich wird der Begriff 'Wiederverwendung' im Zusammenhang mit Wissensdokumenten als wichtiger Aspekt dieser Dissertation definiert.

2.2.1 Wissensdokumente, Lernobjekte und Lernressourcen

Die Definition des Begriffs Wissensdokument ergibt sich aus den Teilen des Kompositums: Wissen und Dokument. Aus informationstechnischer Sicht sind hierbei für den Begriff "Wissen" verschiedene, einander ähnliche Sichtweisen verbreitet. Meistens handelt es sich dabei um stufenweise hierarchische Anordnungen von Begriffen. Eine verbreitete Sicht leitet auf diese Weise einen Zusammenhang zwischen Zeichen, Daten, Information und Wissen her [Bae98]. Dabei unterscheiden sich Daten von Zeichen durch eine für Daten geltende Syntax. Information fügt zu Daten Bedeutung hinzu, während Wissen Information in Handlungen übertragbar macht [Faa04]. Veranschaulicht wird diese Sichtweise durch Abbildung 2.2 anhand eines Verlaufs des Börsenindex DAX. Auf der untersten Ebene finden sich einzelne Zeichen, die ohne eine bestimmte Syntax nicht interpretiert werden können. Kommt die Syntax hinzu, ergeben sich Daten. In diesem Fall die Zuordnung einer Uhrzeit zu einer Dezimalzahl. Auf der nächsten Stufe wird den Daten eine Bedeutung hinzugefügt. Im genannten Beispiel ist das die Tatsache, dass es sich bei den Zahlenpaaren um einen DAX-Verlauf handelt. Wissen wiederum macht diese Information über den DAX-Verlauf übertragbar in konkrete Handlungen. Dies setzt zum einen Hintergrundwissen voraus, wie z.B. dass es sich beim DAX um einen Mittelwert des Wertes verschiedener Aktien handelt, zum anderen aber auch konkretes Handlungswissen, wie man sich bei einem bestimmten Verlauf des DAX verhalten sollte ("Depot muss umgeschichtet werden").

Es gibt ähnliche Ansätze, in denen eine weitere Ebene ("Weisheit") eingeführt wird. So werden Daten, Informationen, Wissen und Weisheit (Data, Information, Knowledge, Wisdom) auf ähnliche Art und Weise in Bezug gesetzt (siehe [DP99]). In den Informationswissenschaften wird der Begriff 'Wissen' wiederum anders definiert. Kuhlen definiert Wissen als durch Semantik verknüpfte Daten, die erst durch Aktion zu Information werden. Information ist laut Kuhlen folglich "Wissen in Aktion" [Kuh04]. Nonaka und Takeuchi, die als Begründer des Wissensmanagements gelten, teilen Wissen in implizites, also nur im Kopf eines Menschen vorhandenes, und explizites, also in explizierter Form vorhandenes Wissen ein [NT95]. Hieraus leiten sie ein Spiralmodell her, das beschreibt, wie implizites Wissen expliziert und für andere nutzbar gemacht werden kann. Eine Aufarbeitung von Definitionen des Wissensbegriffs in der Literatur ist nicht Ziel dieser Arbeit. Hier dient der Begriff *Wissen* allein dazu, ein Verständnis für die im

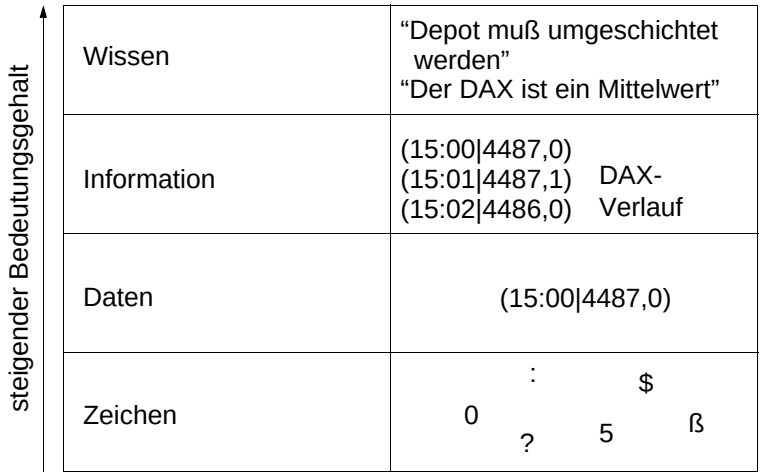

Abbildung 2.2.: Stufenweise Herleitung des Wissensbegriffs [Faa04]

Fokus der vorliegenden Arbeit stehenden Dokumente zu erzeugen und eine einheitliche Nomenklatur zu schaffen. Hierfür ist die gewählte Definition des Wissensbegriffs nach Baets ausreichend. Der Begriff Wissensdokument ist wie folgt definiert:

Definition 2.1 *Ein **Wissensdokument** ist ein elektronisches Dokument, das Wissen enthält.*

Weiterhin haben Wissensdokumente, wie sie in dieser Arbeit betrachtet werden, folgende Eigenschaften:

- Sie sind oftmals textbasiert, können jedoch beliebige multimediale Inhalte (wie zum Beispiel Bilder, Animationen oder Videos) enthalten.
- Sie haben ein hohes Potential, wiederverwendet zu werden.
- Es handelt sich um dynamische Dokumente, die häufig verwendet oder bearbeitet werden.
- Es handelt sich um lokal verwaltete Dokumente.

Zu Wissensdokumenten zählen also beispielsweise Vorlesungs- oder Vortragsfolien, Projektberichte, Skripte aber auch Anforderungskataloge oder Dokumentationen. Dokumente wie zum Beispiel Preis- und Telefonlisten, Anträge oder Fahrpläne zählen nach der oben genannten Definition nicht zu Wissensdokumenten, da sie im Gegensatz zu Wissen, nach Baets, lediglich Daten oder Informationen enthalten. Sie werden nicht generell von der Betrachtung ausgeschlossen, sind aber naturgemäß nicht von großem Interesse für diese Arbeit, da es sich bei solchen Dokumenten zumeist um eher statische Dokumente handelt. Sie ändern sich in geringer Frequenz und weisen kein großes Potential für Wiederverwendung auf. Dadurch ergeben sich während des Lebenszyklus solcher Dokumente kaum Informationen, die im Sinne dieser Arbeit erfasst werden können. Webseiten können zwar auch der Definition für Wissensdokumente genügen, stehen aber nicht im Fokus dieser Arbeit.

Da es im E-Learning-Bereich verwandte Definitionen für die Begriffe ’Lernobjekt’ und ’Lernressource’ gibt, werden diese im Folgenden betrachtet und von Wissensdokumenten abgegrenzt. Die IEEE definiert Lernobjekte wie folgt: "*A learning object is defined as any entity, digital or non-digital, that may be used for*

learning, education or training." [IEE02] Demnach ist ein Lernobjekt ein Objekt, digital oder nicht, das fürs Lernen verwendet werden kann. Diese Definition wird zu Recht von Wiley als zu breit kritisiert, da auch nicht-digitale Objekte von dieser Definition eingeschlossen werden. Er schränkt die Definition wie folgt auf digitale Objekte ein: "*Any digital resource that can be reused to support learning*". [Wil02] Jede digitale Ressource, die zur Unterstützung des Lernens wiederverwendet werden kann, ist demnach ein Lernobjekt. Wiley stellt somit Wiederverwendung schon als wichtigen Prozess im Umfeld von Lernobjekten dar.

Polansi rückt die Wiederverwendbarkeit von Lernobjekten noch weiter in den Vordergrund [Pol03]: "*A Learning Object is an independent and self-standing unit of learning content that is predisposed to reuse in multiple instructional contexts.*" Er fügt die Anforderungen der Unabhängigkeit und Eigenständigkeit hinzu, was den Fokus noch weiter einengt. Somit fallen beispielsweise Webseiten oder Bilder nicht unter den Begriff Lernobjekt. Als eine Eigenschaft von Lernobjekten sieht Polansi auch die Wiederverwendbarkeit in verschiedenen Kontexten. Damit definiert er Lernobjekte als unabhängig vom Kontext, für den sie erstellt wurden, wiederverwendbar. Dies umzusetzen, erweist sich in der Praxis als komplizierte Herausforderung [Hoe05]. Oft ist vielmehr eine Anpassung des jeweiligen Lernobjekts an den neuen Einsatzkontext notwendig [RBH+05]. Daher führen Rensing et al. den Begriff Lernressource als "*digital resource used for e-learning*", also eine digitale Ressource, die fürs E-Learning verwendet wird, ein [RBH+05]. Wie man sieht, variieren die unterschiedlichen Definitionen von Lernobjekten und Lernressourcen sehr stark. Außerdem zeigt die Entwicklung, dass insbesondere der Begriff Lernobjekt immer stärker mit gewissen Eigenschaften, wie Kontextfreiheit oder Wiederverwendbarkeit, belegt wurde. Des Weiteren wird bei beiden Begriffen häufig ein Einhalten etablierter Standards wie Learning Object Metadata (LOM) [IEE02] oder Shareable Content Object Reference Model (SCORM) [ADL04] oder ein bestimmtes Format (wie Web Based Training) impliziert. Um dieser Vorbelegung auszuweichen, wurde entschieden, den Begriff Wissensdokument für den Typ von Dokumenten, die im Fokus dieser Arbeit stehen, einzuführen. In der Regel können jedoch auch Lernressourcen und - abhängig von der verwendeten Definition - Lernobjekte zu den Wissensdokumenten gezählt werden.

2.2.2 Autorenprozesse und Wiederverwendung

Für eine Erfassung von Informationen aus dem Lebenszyklus eines Dokuments sind vorrangig die Prozesse, die während seines Lebenszyklus ausgeführt werden, von Interesse. Autorenprozesse, die auf einem Dokument ausgeführt werden, werden im Folgenden gesondert betrachtet, da hier, besonders im E-Learning-Bereich, zahlreiche Vorarbeiten existieren, die ggf. auf Wissensdokumente abgebildet werden können. Andere Arten von Prozessen, wie beispielsweise betriebswirtschaftliche Prozesse, werden hier nicht berücksichtigt. Im Rahmen dieser Arbeit sind nur Prozesse relevant, die auf dem Dokument selbst ausgeführt werden. Auch das Verbinden von Wissen und Prozessen, bei dem ein vorhandener Prozess mit Wissensobjekten annotiert wird, um den Prozess selbst zu dokumentieren, wie es beispielsweise bei "Wissensbasiertem und Prozessorientiertem Innovationsmanagement" (WPIM) [VH07] angewendet wird, ist nicht Bestandteil dieser Arbeit.

Autorenprozesse

Im E-Learning-Bereich wurden in den vergangenen Jahren verschiedene Autorenprozesse zur Unterstützung von Wiederverwendung oder zur Steigerung der Effizienz beim Erstellen von Lerninhalten entwickelt und entsprechende Werkzeuge dafür implementiert. Die Prozesse werden im Folgenden erläutert und auf ihre Übertragbarkeit auf Wissensdokumente hin überprüft. Hierzu zählen Re-Authoring, Re-Purposing und Authoring-by-Aggregation.

Laut [RZM+08] ist unter *Re-Authoring* das Modifizieren einer bestehenden Lernressource zum Zwecke der Wiederverwendung zu verstehen. Darunter fallen Fehlerkorrekturen und Aktualisierungen ebenso wie Re-Purposing. Re-Purposing ist also ein Unterbegriff von Re-Authoring. Durch *Re-Purposing* wird eine Lernressource einem neuen Einsatzkontext entsprechend angepasst. Ein Beispiel wäre eine Lernressource, die auf eine neue Zielgruppe ausgerichtet werden muss. Dies hat inhaltliche Auswirkungen, z.B. auf die verwendete Terminologie. Um diese Anpassung zu erreichen, müssen Lernressourcen häufig in Komponenten zerlegt, diese entsprechend angepasst und wieder zusammengeführt werden. Daher vereinigt der Begriff Re-Purposing die Prozesse Modularisierung [MRS06], Aggregation [VJGD05] und Anpassung [Zim08]. Für die Anpassung eines Dokuments gibt es verschiedene Möglichkeiten, die in [ZRS06] erläutert werden. Hierzu zählen zum Beispiel die Überführung eines Dokuments in ein neues Ausgabeformat, Änderung der Terminologie, der Zielgruppe oder des Layouts. Rensing et al. betrachten das Ergebnis des Re-Authoring-Prozesses als Teil der logischen Ressource, die dem Re-Authoring-Prozess unterzogen wurde, auch wenn sie physikalisch in mehreren Ausprägungen vorhanden ist [RBH+05]. Andere sehen das Ergebnis eines Re-Authoring-Prozesses als neue Lernressource mit neuem Lebenszyklus an [CVB06]. Unter *Authoring-by-Aggregation* wird das Aggregieren verschiedener kleinerer Lernressourcen zu einer neuen Lernressource verstanden. Dabei können nach Hoermann auch Änderungen an den aggregierten Ressourcen vorgenommen oder neue Teile eingefügt werden [Hoe05]. Eine frühe Definition des Begriffs schließt diese Möglichkeiten noch aus [DH03]. Meyer erweitert den Authoring-by-Aggregation-Prozess in [Mey08] um eine zusätzliche vorgeschaltete Design-Phase.

Es wurden verschiedene Werkzeuge zur Unterstützung der genannten Prozesse entwickelt. In [MZRS07] wird ein Werkzeug zur Unterstützung von Modularisierung beschrieben. [MHRS06] beschreibt ein Framework zur Unterstützung von Re-Purposing, während in [ZRS07] ein Werkzeug zur Unterstützung von Anpassungsprozessen vorgestellt wird. Die Plattform docendo (ehemals 'Resource-Center') [HHRS05] unterstützt Authoring-by-Aggregation. Die meisten dieser Werkzeuge bauen auf verbreiteten E-Learning-Standards wie LOM oder SCORM auf oder setzen diese voraus. Für die Verarbeitung von Wissensdokumenten sind sie daher nur in sehr geringem Umfang geeignet.

Die beschriebenen Autorenprozesse selbst sind zwar auch für Wissensdokumente denkbar und werden teilweise auch auf Wissensdokumenten durchgeführt, es gibt jedoch kaum Werkzeuge, die diese Prozesse für Wissensdokumente im Allgemeinen unterstützen. Deshalb haben diese Prozesse und Werkzeuge hier kaum praktische Relevanz. Darum ist eine Verwendung der beschriebenen Begrifflichkeiten im Kontext von Wissendokumenten weniger sinnvoll. Es wird daher allgemeiner definiert:

Definition 2.2 *Unter **Überarbeitung** eines Wissensdokuments versteht man ein erneutes Bearbeiten eines Wissensdokuments. Dabei können Inhalte aus verschiedenen Wissensdokumenten zusammengeführt, einzelne Teile wiederverwendet oder das Dokument an einen neuen Einsatzkontext angepasst werden.*

Dieser Begriff subsummiert die oben beschriebenen Autorenprozesse. Da Wiederverwendung ein häufiger und wichtiger Schritt bei der Überarbeitung eines Dokuments ist und sie in dieser Arbeit eine große Bedeutung hat, wird im Folgenden das dieser Arbeit zu Grunde liegende Verständnis von Wiederverwendung erläutert.

Wiederverwendung

Für Lernressourcen ist Wiederverwendung von Rensing et al. definiert als "erneute Verwendung einer bestehenden Lernressource, die bereits in einem bestimmten Kontext verwendet wird". Sie kann ohne oder mit Änderungen an der Ressource erfolgen [RBH+05]. Diese Definition ist auch auf Wissensdokumente übertragbar, ist jedoch für die Zwecke dieser Arbeit zu allgemein gehalten.

Libbrecht stellt in [Lib08] ein detaillierteres Modell für Wiederverwendung von E-Learning Inhalten vor. Er unterteilt dazu die aktuelle Praxis der Wiederverwendung in fünf Kategorien:

1. **Unveränderte Übernahme (Verbatim Inclusion)**: Diese Praxis entspricht einer unveränderten Übernahme der Inhalte. Am häufigsten tritt dies bei Medienobjekten wie Bildern auf. Aber auch komplette Dokumente können auf diese Art wiederverwendet werden.

2. **Kopieren und Einfügen (Copy-and-Paste)**: Hierbei werden einzelne Teile eines Lernobjekts kopiert und in ein neues oder bestehendes Lernobjekt eingefügt.

3. **Kopieren und Überarbeiten (Copy-and-Branch of large bodies)**: Hierbei wird ein bestehendes Lernobjekt kopiert, verändert und als neues Lernobjekt gespeichert. Das hat eine neue Version dieses Lernobjekts mit geändertem Inhalt zur Folge. Diese Form der Wiederverwendung bietet sich besonders dann an, wenn ein großer Teil eines bestehenden Lernobjekts wiederverwendet werden kann.

4. **Verknüpfen (Linking)**: Das Einfügen einer Referenz (zum Beispiel in Form eines Hyperlinks) auf andere Inhalte führt zu dieser Form der Wiederverwendung. Referenzieren reflektiert Erweiterungen oder Aktualisierungen an den verlinkten Inhalten. Wenn ein referenzierter Inhalt jedoch nicht mehr vorhanden sein sollte oder sich die Lokation ändert, zeigt die Referenz ins Leere.

5. **Kanalbasierte Wiederverwendung (Channel-based)**: Unter kanalbasierter Wiederverwendung versteht Libbrecht die Aggregation und Verschmelzung verschiedener Feeds, um neue Inhalte zu erzeugen. Ein Beispiel hierfür sind Nachrichtenseiten, die verschiedene Newsfeeds zusammenführen.

Die größte Relevanz in Bezug auf Wissensdokumente haben die ersten drei Kategorien. Sie stellen gleichzeitig auch die häufigsten Formen der Wiederverwendung in diesem Bereich dar. Anhand des Beispiels einer digitalen Folienpräsentation können die verschiedenen Formen der Wiederverwendung verdeutlicht werden. Einer unveränderten Übernahme entspricht zum einen die Wiederverwendung einer kompletten Präsentation in unveränderter Form, aber zum anderen auch das Einfügen einer bestehenden Bild-, Audio- oder Videodatei in die Präsentation. Kopieren und Einfügen erfolgt beispielsweise, wenn einzelne Folien, Text oder sonstige Inhalte von einer Präsentation in eine andere kopiert werden. Kopieren und Überarbeiten wäre der Fall, wenn eine bestehende Präsentation geändert und unter neuem

Namen gespeichert wird. Es ist auch eine Verknüpfung von Inhalten möglich, indem z.B. Hyperlinks in ein Dokument eingepflegt oder Medienobjekte per Referenz eingefügt werden. Eine kanalbasierte Wiederverwendung findet bei lokalen Wissensdokumenten typischerweise nicht statt.

2.3 Der Lebenszyklus von Lernobjekten und Wissensdokumenten

Um Lebenszyklusinformationen von Dokumenten erfassen zu können, ist es notwendig, den Lebenszyklus der Dokumente zu modellieren. Dann können die Prozesse, die zur Entstehung von Lebenszyklusinformationen führen, sowie die entsprechenden Systeme und Applikationen, in denen die Informationen entstehen, identifiziert werden. Im folgenden Abschnitt werden zunächst existierende Lebenszyklusmodelle aus dem E-Learning- und Dokumentenmanagement-Bereich analysiert und auf eine Übertragbarkeit auf Wissensdokumente hin überprüft. Aus den Ergebnissen wird schließlich ein Modell für den Lebenszyklus von Wissensdokumenten entwickelt.

2.3.1 Bestehende Lebenszyklusmodelle

Es gibt bereits verschiedene Ansätze zur Modellierung des Lebenszyklus von Lernobjekten, Lernressourcen und Wissensdokumente, die für unterschiedliche Zwecke entworfen wurden. Die wichtigsten hiervon werden im Folgenden vorgestellt.

Auch im betriebswirtschaftlichen Bereich gibt es diverse Anwendungsszenarien, für die Lebenszyklusmodelle entworfen und umgesetzt wurden. Das bekannteste ist hierbei das Produktlebenszyklusmanagement (PLM) [Sen09]. Dabei handelt es sich um einen strategischen Gesamtansatz zum Management eines bestimmten Produktes. Als Basis hierfür wird der Lebenszyklus eines Produkts vom Entwurf über die Konstruktion und Produktion bis hin zu Dienstleistungen modelliert. Ein solcher Lebenszyklus kann auch den Lebenszyklus eines Dokuments im weiteren Sinne widerspiegeln, zum Beispiel wenn es sich bei dem Produkt um ein Dokument, etwa einen E-Learning-Kurs, handelt, wie im Projekt EXPLAIN beschrieben [ZBC+05]. Bei PLM geht es vorrangig darum die Prozesse, die zur Entwicklung eines Produkts notwendig sind zu identifizieren und zu unterstützen. Diese decken sich jedoch nicht mit Prozessen, die auf einem Dokument selbst während dessen Lebenszyklus ausgeführt werden.

Strijkers Modell

Strijker analysiert in [Str04] den traditionellen Lebenszyklus von Lernobjekten. Dieser besteht aus sechs aufeinanderfolgenden Phasen, die in Abbildung 2.3 dargestellt sind.

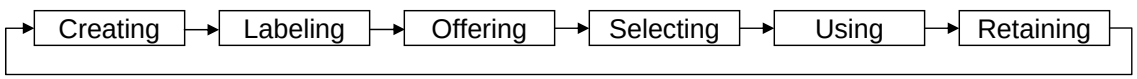

Abbildung 2.3.: Der traditionelle Lebenszyklus von Lernobjekten nach Strijker [Str04]

Die von Strijker identifizierten Phasen sind die Folgenden:

1. **Erstellung (Creating)**: Ein Lernobjekt wird erstellt. Dies kann entweder durch Erstellen eines komplett neuen Lernobjektes oder durch Wiederverwendung eines bereits Bestehenden erfolgen. Das wiederverwendete Lernobjekt muss jedoch, laut Strijker, für die Wiederverwendung konzipiert sein.

2. **Auszeichnung (Labeling)**: In dieser Phase wird das Lernobjekt mit Metadaten versehen, was vor allem der Auffindbarkeit dient.

3. **Angebot (Offering)**: In der Angebotsphase wird das Lernobjekt, beispielsweise in einem Lern-Content-Management-System (LCMS) oder Lernobjekt-Repository (LOR), angeboten.

4. **Auswahl (Selecting)**: Das Lernobjekt wird von einem Lehrenden ausgewählt. Hierfür muss es auffindbar, also zum Beispiel mit entsprechenden Metadaten ausgezeichnet sein.

5. **Nutzung (Using)**: In dieser Phase erfolgt die Nutzung eines Lernobjektes. Wenn das Lernobjekt den Anforderungen des Lehrenden oder Autoren genügt, kann es direkt wiederverwendet werden, ansonsten muss es geändert werden, was in dieser Phase implizit ist.

6. **Aufbewahrung (Retaining)**: Nach erfolgter Nutzung kann das Lernobjekt entweder so belassen werden wie es ist, den Erfahrungen aus der Nutzung entsprechend angepasst werden oder aber ganz aus dem Angebot entfernt werden, weil es beispielsweise veraltet ist.

Die Abfolge der Phasen in diesem Modell ist streng sequentiell, was zur Folge hat, dass eine Phase nur über die vorhergehende erreicht werden kann. In einer weiteren Ausprägung des Modells ermöglicht Strijker alternative Pfade durch die unterschiedlichen Phasen des Lebenszyklus. Dies entspricht eher der Praxis, da oftmals Phasen, wie Auszeichnung oder Auswahl, übersprungen werden, wenn der Autor gleichzeitig ein Nutzer des Lernobjektes ist.

Strijkers Lebenszyklusmodell ist auf die These ausgerichtet, dass Lernobjekte oftmals so wie sie sind oder mit lediglich kleinen Änderungen wiederverwendet werden können. Anpassungen werden nur als Bestandteil der Nutzungsphase berücksichtigt. Es hat sich jedoch erwiesen, dass dies nicht der Praxis entspricht [RBH+05]. Wiederverwendung findet insgesamt zu wenig Bedeutung in diesem Modell. Des Weiteren geht dieses Modell von einer expliziten Auszeichnungsphase zur meist manuellen Erstellung von Metadaten aus, was der Idee der automatisierten Erfassung von Informationen während des gesamten Lebenszyklus eines Dokuments widerspricht.

Cardinaels Modell

Cardinaels kritisiert Strijkers Modell ebenfalls in diesen Punkten: Der Aspekt der Wiederverwendung kommt zu wenig zur Geltung, da eine Wiederverwendung nur als Bestandteil der Nutzungsphase erfolgen kann und eine Auszeichnung der Lernobjekte lediglich in der dafür vorgesehenen expliziten Auszeichnungsphase erfolgt. Er schlägt ein dynamisches Lebenszyklusmodell vor, welches ein stärkeres Gewicht auf Wiederverwendung und die kontinuierliche Gewinnung von Metadaten legt [Car07]. Es setzt sich aus folgenden sieben Phasen zusammen (siehe Abbildung 2.4).

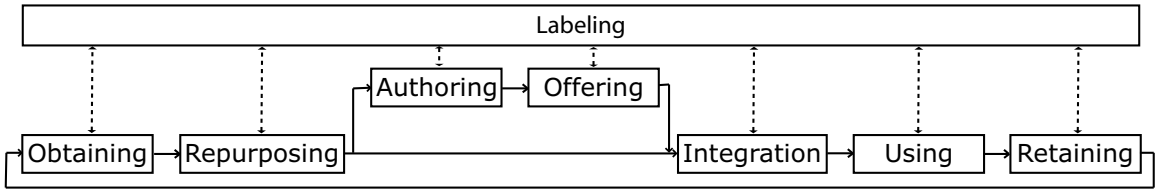

Abbildung 2.4.: Cardinales dynamischer Lebenszyklus für Lernobjekte [Car07]

1. **Bezug (Obtaining)**: Cardinaels setzt den Bezug der Lernobjekte an den Anfang des Lebenszyklus, um den Aspekt der Wiederverwendung zu stärken. Hier beginnt der Lebenszyklus eines Lernobjekts mit der Suche nach bereits vorhandenen Lernobjekten, die gegebenenfalls wiederverwendet werden können.

2. **Re-Purposing (Repurposing)**: Diese Phase bereitet das Lernobjekt auf die nachfolgenden Phasen vor. Nach Cardinaels Verständnis wird ein Lernobjekt dabei nur geringfügig modifiziert. Hiernach wird entschieden, ob es direkt genutzt werden kann, verändert werden muss oder als Teil eines neuen Lernobjekts wiederverwendet wird. Zu Re-Purposing zählen beispielsweise Aggregation oder Disaggregation (Modularisierung).

3. **Bearbeitung (Authoring)**: Diese Phase deckt sowohl das Erstellen eines neuen Lernobjektes, als auch die Anpassung eines bereits existierenden Lernobjekts ab. Ist eine direkte Wiederverwendung eines Lernobjektes möglich, kann diese Phase übersprungen werden.

4. **Angebot (Offering)**: Entsprechend Strijkers Modell werden in dieser Phase die Lernobjekte angeboten, beispielsweise in einem Repository oder LCMS.

5. **Einbindung (Integration)**: Die Lernobjekte werden in den Lernkontext der Lernenden eingebunden.

6. **Nutzung (Using)**: Diese Phase spiegelt die eigentliche Nutzung durch Lernende wider.

7. **Aufbewahrung (Retaining)**: Diese Phase deckt - wie in Strijkers Modell - den Rest des Lebenszyklus eines Lernobjektes nach erfolgter Nutzung ab.

8. **Auszeichnung (Labeling)**: Die Auzeichnungsphase wird von Cardinaels als implizite Phase, die während des gesamten Lebenszyklus aus allen anderen Phasen heraus erreichbar ist, modelliert. Hierdurch entsteht die Möglichkeit, zu jedem Zeitpunkt im Lebenszyklus Metadaten zu einem Lernobjekt zu gewinnen.

Dadurch, dass die Auszeichnung parallel zu den anderen Phasen geschieht, ist diese Phase bei Cardinaels weniger eine explizite Phase, sondern vielmehr ein Aspekt, der in jeder der anderen Phasen in Betracht gezogen werden kann und sollte. Die Metadaten eines Lernobjekts werden so zu einer dynamischen Instanz, die sich auch während des Lebenszyklus ändern und an neue Gegebenheiten angepasst werden kann.

Das Modell von Cardinaels ist, besonders hinsichtlich der Fokussierung auf Wiederverwendung sowie der Art und Weise der Modellierung der Auszeichnungsphase, für Wissensdokumente interessant.

Wiederverwendung wird dem Lebenszyklus eines Lernobjektes durch die Bezugsphase als Einstieg in den Lebenszyklus zu Grunde gelegt. Für die Modellierung des Lebenszyklus von Wissensdokumenten zum Zweck der Gewinnung von Lebenszyklusinformationen ist dieses Modell, wie auch das von Strijker, jedoch zu komplex. Einige der modellierten Phasen haben für die Gewinnung von Lebenszyklusinformationen, worauf der Fokus dieser Arbeit liegt, und auch für Wissensdokumente im Allgemeinen keine Relevanz oder können zusammengefasst werden.

Lebenszyklusmodell für Lernressourcen nach Rensing et al.

Rensing et al. schlagen in [RZM+08] ein deutlich einfacheres Modell für den Lebenszyklus von Lernressourcen vor. Das beschriebene Modell beschränkt sich auf vier Phasen:

1. **Erstellung (Authoring)**: Es erfolgt die Erstellung einer Lernressource - oftmals für einen bestimmten Kontext.
2. **Bereitstellung (Provision)**: Nach Erstellung wird eine Lernressource zumeist bereitgestellt. Dies kann beispielsweise in Repositories oder Marktplätzen erfolgen.
3. **Lernen (Learning)**: In der Lernphase erfolgt die Nutzung der Lernressource durch Lernende oder Lehrende.
4. **Re-Authoring (Re-Authoring)**: Hier kann die Lernressource geändert und einem neuen Einsatzkontext entsprechend angepasst werden (siehe 2.2.2).

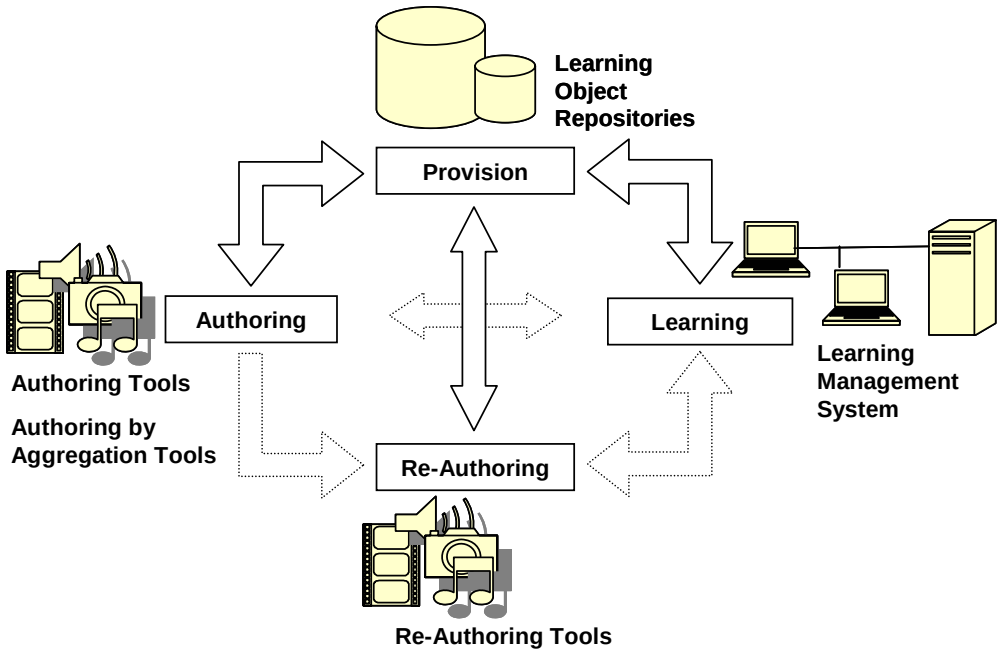

Abbildung 2.5.: Lebenszyklusmodell für Lernressourcen nach Rensing et al. [RBH+05]

Die Phasen 1-3 stellen für die Autoren das traditionelle Lebenszyklusmodell für Lernressourcen dar. Sie führen Phase 4 als explizite Re-Authoring Phase in den Lebenszyklus ein, um der Tatsache gerecht

zu werden, dass Lernressourcen oftmals an den neuen Einsatzkontext angepasst werden müssen, bevor sie wiederverwendet werden können. Das Modell sieht weiterhin Querverbindungen zwischen den Phasen vor. So können beispielsweise die Authoring wie auch die Re-Authoring Phase von der Bereitstellungsphase aus erreicht werden, was einer Suche nach bestehenden Lernressourcen zum Zweck der Wiederverwendung entspricht.

Dieses Modell ist sehr kompakt. Wiederverwendung wird durch eine explizite Re-Authoring Phase modelliert. Eine Auszeichnung mit Metadaten ist jedoch weder als einzelne Phase noch als kontinuierlicher Vorgang Teil des Modells. Zudem kommt ein reines Lernen als Nutzungsphase für Wissensdokumente nicht in Frage.

Für alle bisher vorgestellten Modelle gilt, dass sie stark auf Lernressourcen bzw. Lernobjekte ausgerichtet und in ihrer Form nicht auf Wissensdokumente übertragbar sind. Zumeist fehlt die Berücksichtigung des Wiederverwendungsaspekts oder die Auszeichnung mit Metadaten und Gewinnung von Informationen ist als explizite Phase und nicht als kontinuierlicher Prozess vorgesehen. In den meisten Fällen sind die existierenden Modelle zudem zu komplex. Im Folgenden werden Lebenszyklusmodelle aus dem Bereich des Dokumenten- und Informationsmanagements darauf hin untersucht, ob sie als Basis für ein Modell für Wissensdokumente in Frage kommen.

Lebenszyklusmodell für Dokumente nach Ginsburg

In [Gin99] stellt Ginsburg ein Lebenszyklusmodell für Intranet-Dokumente vor. Es handelt sich hierbei um ein Wasserfallmodell, welches aus 5 Phasen besteht (vergl. Abbildung 2.6):

1. **Erstellung (Creation)**: Da das Modell für Dokumente aus dem Intranet eines Unternehmens ausgelegt ist, findet die Erstellung der Dokumente durch Autoren auf lokalen Rechnern statt. Die Erstellungphase enthält auch die Auszeichnung des Dokuments mit Metadaten.

2. **Veröffentlichung (Publication)**: Nach Erstellung werden die Dokumente auf einem Server zur Verfügung gestellt. Diese Phase beinhaltet auch die Umwandlung in ein anderes Dokumentformat (zum Beispiel von Microsoft Word nach PDF).

3. **Organisation (Organization)**: In dieser Phase wird das Dokument in die lokale Hierarchie des Servers eingeordnet - dies kann auch mit Hilfe von Ontologien geschehen. Desweiteren werden Verknüpfungen, die auf dieses Dokument verweisen, erstellt bzw. aktualisiert.

4. **Zugriff (Access)**: Die Zugriffsphase deckt die Interaktion mit einem Nutzer ab, der nach einem bestimmten Dokument im System sucht. Hierzu zählen demnach die Auswahl von Suchbegriffen, das Absetzen einer Suchanfrage, die Auswahl eines Suchergebnisses und das Nutzen (Lesen) eines Dokuments. Auch das Hinzufügen von weiteren Metadaten zum Dokument und das Abgeben einer Rückmeldung über den Suchvorgang zählt Ginsburg zu dieser Phase.

5. **Zerstörung (Destruction)**: Diese Phase umfasst das Überschreiben oder Löschen eines Dokuments.

Die Auszeichnung eines Dokuments mit Metadaten findet in diesem Modell kaum Beachtung. Ginsburg weist darauf hin, dass eine Auszeichnung eine Vorabinvestition des Unternehmens erfordert. Außerdem

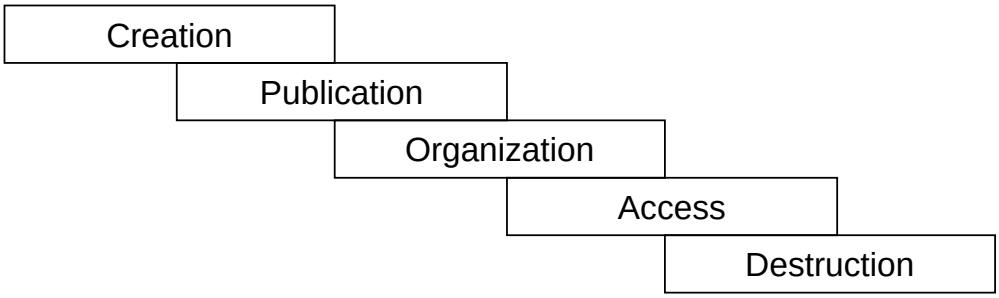

Abbildung 2.6.: Lebenszyklusmodell nach Ginsburg

befinden sich gewöhnlich große Mengen an unstrukturierten und nicht ausgezeichneten Dokumenten in den Intranets von Unternehmen, was eine Handhabung weiter erschwert. Das Modell ist insgesamt sehr auf die Verwaltung von Dokumenten im internen Netzwerk eines Unternehmes ausgerichtet. Darauf deutet beispielsweise eine Trennung von "Veröffentlichung" und "Organisation" in unterschiedliche Phasen hin. Wiederverwendung wird in diesem Modell überhaupt nicht berücksichtigt. Das im Modell verwendete Vokabular ist jedoch eher geeignet als das der Lebenszyklusmodelle aus dem E-Learning-Bereich, da es auf allgemeine Dokumente und nicht auf Lernobjekte ausgerichtet ist.

SHAMAN-Lebenszyklusmodell für archivierte digitale Objekte

Brocks et al. stellen in [BKJH10] ein Lebenszyklusmodell für digitale Objekte vor. Es im Rahmen des SHAMAN-Projektes (**S**ustaining **H**eritage **A**ccess through **M**ultivalent **A**rchivi**N**g) entwickelt. In SHAMAN geht es um die Entwicklung eines neuartigen Frameworks für die Langzeitarchivierung. Das entiwckelte Lebenszyklusmodell (siehe Abbildung 2.7) besteht aus folgenden fünf Phasen:

1. Erzeugung (Creation): In dieser Phase wird ein digitales Objekt erzeugt. Die Prozesse hierfür können sehr komplex sein und diverse Aktoren daran beteiligt sein, bevor Informationen entstehen, die es wert sind archiviert zu werden. Die Nutzung der digitalen Objekte fällt ebenso in diese Phase.

2. Zusammenstellung (Assembly): Diese Phase umfasst die Begutachtung des digitalen Objektes sowie die für seine Archivierung benötigten Verarbeitungsschritte.

3. Archivierung (Archival): Diese Phase umfasst den Zeitraum, während dessen sich das digitale Objekt im Archiv befindet.

4. Anpassung (Adoption): In dieser Phase wird ein Objekt für eine spätere Wiederverwendung angepasst und integriert.

5. Wiederverwendung (Reuse): Hier wird ein digitales Objekt, das aus dem Archiv bezogen wurde, genutzt. Bei Wiederverwendung können neue Metadaten entstehen, die wiederum dem Objekt im Archiv zugeführt werden müssen.

Das hier vorgestellte Lebenszyklusmodell beschreibt den Lebenszyklus eines digitalen Objektes aus der Perspektive der Langzeitarchivierung. Aus dieser Perspektive sind die Erstellung und die Nutzung gleich-

zeitig stattfindende Phasen im Lebenszyklus [BKJH10], werden also auch nicht getrennt betrachtet und modelliert. Für das angestrebte Lebenszyklusmodell von Wissensdokumenten sind diese Phasen jedoch zu unterscheiden. Die Erfassung von Metadaten während des Lebenszyklus von digitalen Objekten soll in SHAMAN ebenfalls adressiert werden (siehe Kapitel 3.1) und ist somit auch dem Lebenszyklusmodell inhärent. In dieser Form ist der vorgestellte Lebenszyklus für Wissensdokumente jedoch aus genannten Gründen nicht für direkt verwendbar.

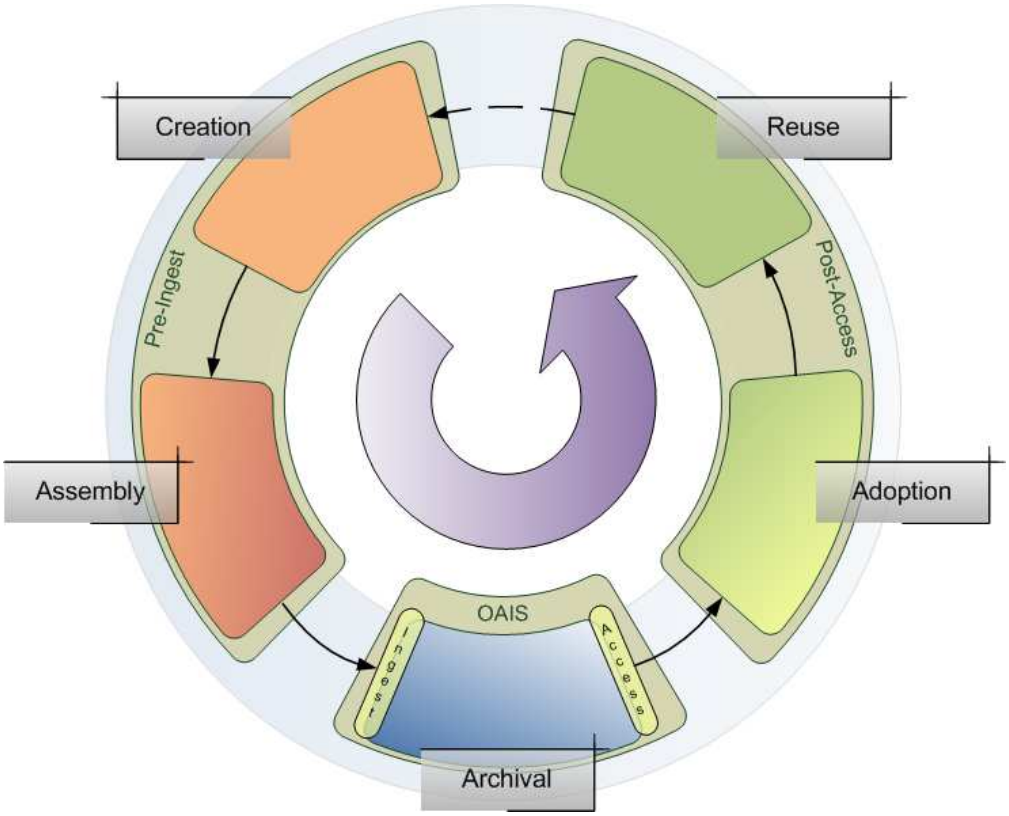

Abbildung 2.7.: Shaman-Lebenszyklusmodell nach Brock et al. [BKJH10]

Weitere Lebenszyklusmodelle aus dem Informations- und Dokumentenmanagement

Im Dokumentenmanagement werden in Bezug auf Informationsressourcen typischerweise folgende Lebenszyklusphasen unterschieden [MHP09]:

1. Erzeugung (Creation): Hier werden Informationsressourcen erzeugt. Dies kann auch unter Wiederverwendung anderer Ressourcen geschehen.
2. Überarbeitung (Revision): In dieser Phase werden Informationsressourcen geändert, erweitert, annotiert oder korrigiert. Dies kann beliebig oft durch beliebig viele Autoren geschehen.
3. Stabilität (Stability): Wenn eine Informationsressource diese Phase erreicht hat, unterliegt sie keinen häufigen Änderungen mehr. Der Übergang in diese Phase kann auch durch einen formalen Prozess, wie zum Beispiel eine Freigabe erfolgen.

4. Archivierung (Archival): Eine Achivierung erfolgt, wenn die Ressourcen nicht mehr aktiv genutzt werden, aber dennoch, beispielsweise aus rechtlichen Gründen, für einen gewissen Zeitraum aufbewahrt werden müssen.

5. Löschen (Deletion): Schließlich werden die Ressourcen, wenn sie nicht mehr benötigt werden, gelöscht.

Die Phasen dieses Modells besteht im Vergleich zu den anderen Modellen aus eher statischen Phasen. Die Stabilitäts-Phase wird in dem gewünschten Lebenszyklusmodell für Wissensdokumente nicht benötigt, da in einer solchen Phase keine Lebenszyklusinformationen auftreten. Wiederverwendung ist auch hier Teil der Erstellung einer Informationsressource. Die Nutzung der Ressource selbst wird gar nicht berücksichtigt.

Beim *Informationslebenszyklusmanagements* (ILM) geht es darum, die Speicherung von Information entsprechend ihrem Wert zu modellieren. Dies bedeutet, dass niedrigwertige Informationen auf kostengünstigen Speicherlösungen untergebracht werden, während höherwertige Informationen auf teuren Speicherlösungen abgelegt werden [Tur09]. Auch in diesem Zusammenhang werden Phasen im Lebenszyklus der Informationen definiert [GSMP09]:

1. Erzeugung oder Empfang (Creation and Receipt): Eine Information wird innerhalb einer Organisation erzeugt oder aus einer externen Quelle erhalten.

2. Verteilung (Distribution): Die Information wird organisationsintern oder -extern verteilt.

3. Nutzung (Use): Die Nutzung geschieht nach der Verteilung. Hierdurch können zum Beispiel Geschäftsprozesse angestoßen oder weitere Aktionen dokumentiert werden.

4. Pflege (Maintenance): Hierunter wird die Verwaltung der Information verstanden. Dazu zählt auch das Verfügbar- und Auffindbarmachen der Information, aber auch das Retrieval selbst.

5. Disposition (Disposition): Entspricht weitestgehend der im letzten Abschnitt aufgeführten Archivierungsphase. Die Information wird für eine etwaige Nutzung bereitgehalten. Je länger diese Phase dauert, desto kostengünstiger ist im allgemeinen die Speicherlösung, die zur Ablage der Information dient.

Dieses Modell beschreibt den Lebenszyklus einer Information aus betriebswirtschaftlicher Sicht. Hierbei geht es unter anderem darum einem Dokument anhand seiner Phase im Lebenszyklus einen Wert und somit eine Speicherlösung zuzuordnen. Wiederverwendung oder die Erfassung von Metadaten werden in diesem Modell nicht berücksichtigt.

Ein weiteres Lebenszyklusmodell aus dem Dokumentenbereich wird von Antonacopoulos et al. in [AKKW04] vorgestellt. Hierbei handelt es sich um den Lebenszyklus eines historischen Dokuments von der Papierform, über verschiedene Stufen der Digitalisierung, bis hin zum verarbeitbaren elektronischen Dokument. Es ist eher als Vorgehensmodell bei der Erhaltung und Digitalisierung von Dokumenten in Papierform gedacht, denn als Lebenszyklusmodell. Deshalb eignet es sich weniger als Grundlage für ein Lebenszyklusmodell für Wissensdokumente.

2.3.2 Ein Lebenszyklusmodell für Wissensdokumente

Ein Modell bildet die Wirklichkeit in einer bestimmten Art und Weise ab und dient dabei einem bestimmten Zweck. Eine Modellierung ist gegenüber der Realität grundsätzlich mit einem Informationsverlust verbunden [DSS93]. Somit ist eine Übertragbarkeit eines bereits existierenden Modells auf ein bestimmtes Problem nur dann möglich, wenn das existierende Modell demselben Zweck dient, bzw. das Modell auf das gegebene Problem anwendbar ist. Das angestrebte Lebenszyklusmodell für Wissensdokumente in dieser Arbeit soll folgende Ziele erfüllen:

- Es soll anwendbar sein auf Dokumente, die der in Kapitel 2.2.1 gegebenen Definition genügen.
- Es soll die Identifikation der Systeme und Applikationen, die am Lebenszyklus dieser Dokumente beteiligt sind, ermöglichen.
- Es soll helfen, Informationen, die während des Lebenszyklus anfallen, zu identifizieren, strukturieren und kategorisieren.

Keines der in Kapitel 2.3.1 vorgestellten Lebenszyklusmodelle erfüllt diese Anforderungen vollständig. Einige beinhalten jedoch Aspekte, die für eine Modellierung des Lebenszyklus von Wissensdokumenten berücksichtigt werden sollten. Ein Ziel dieser Arbeit ist die Erfassung von Informationen, die während des Lebenszyklus eines Wissensdokuments anfallen. Eine Möglichkeit zur Erfassung von Informationen aus allen Phasen heraus, wie es im Modell von Cardinaels vorgesehen ist, ist daher notwendig. Aus dem Modell von Cardinales kann auch die Gewichtung der Wiederverwendung im Modell übernommen werden. Wissensdokumente, wie beispielsweise Berichte oder Präsentationen, werden selten komplett neu, sondern häufig auf Basis bereits bestehender Dokumente erstellt. Der Vorteil des Modells von Rensing et al. ist die Kompaktheit. Viele der in den anderen Modellen eingeführten Phasen sind für eine Identifikation von Lebenszyklusinformationen nicht notwendig. Beide Modelle sind jedoch auf Lernobjekte bzw. Lernressourcen ausgerichtet. Einige der genannten Phasen sind im Lebenszyklus von Wissensdokumenten so nicht vorhanden, außerdem muss die Terminologie entsprechend angepasst werden. Das hier vorgeschlagene Modell vereint die Vorteile der Modelle von Cardinaels und Rensing und passt diese auf den neuen Kontext an. Das vorgeschlagene Lebenszyklusmodell gliedert sich daher in folgende Phasen:

- **Bereitstellung / Zugriff**: Die Zugriffsphase steht oft am Anfang des Lebenszyklus eines Wissensdokuments, da der erste Schritt meistens die Suche nach bereits existierenden Wissendokumenten ist, die für eine Wiederverwendung in Frage kommen. Die Bereitstellung stellt hier den umgekehrten Vorgang des Zugriffs dar. Beim Zugriff wird ein Dokument aus einem System erhalten, bei der Bereitstellung wird es in einem System verfügbar gemacht. Da dies häufig in ein und demselben System (z.B. Dateisystem, Dokumentenmanagement-System, Content Management-System) geschieht, wurden diese Vorgänge zu einer Phase vereint.
- **Erstellung / Überarbeitung (Wiederverwendung)**: Diese Phase umfasst sowohl die Erstellung eines komplett neuen Dokuments, als auch die Überarbeitung und dadurch Wiederverwendung eines bestehenden Dokuments. Auch eine Aggregation mehrerer bestehender Dokumente ist durch diese Phase abgedeckt. Im Fall von Wissendokumenten werden in den meisten Fällen dieselben

Applikationen zur Erstellung wie auch zur Überarbeitung genutzt, deshalb ergibt sich eine, diese Prozesse umfassende Phase.

- **Nutzung**: Ein Wissensdokument kann, abhängig von seinem Format, auf unterschiedliche Weise genutzt werden. So kann eine Nutzung beispielsweise durch Präsentation eines Foliensatzes, das Ausdrucken oder Lesen eines Dokuments oder auch das Abspielen in einer entsprechenden Applikation erfolgen.
- **Erfassung von Informationen**: Die Erfassung ist keine explizite Phase des Lebenszyklus, sondern erfolgt aus allen anderen Phasen heraus. Somit können zu jedem Zeitpunkt im Lebenszyklus Informationen über ein Wissensdokument (Lebenszyklusinformationen) gewonnen werden.

Jede Phase des Lebenszyklus kann in jede andere Phase übergehen. So kann beispielsweise eine Nutzung direkt nach der Erstellungsphase erfolgen oder aus der Bereitstellungsphase heraus. Abbildung 2.8 stellt den entworfenen Lebenszyklus für Wissensdokumente dar.

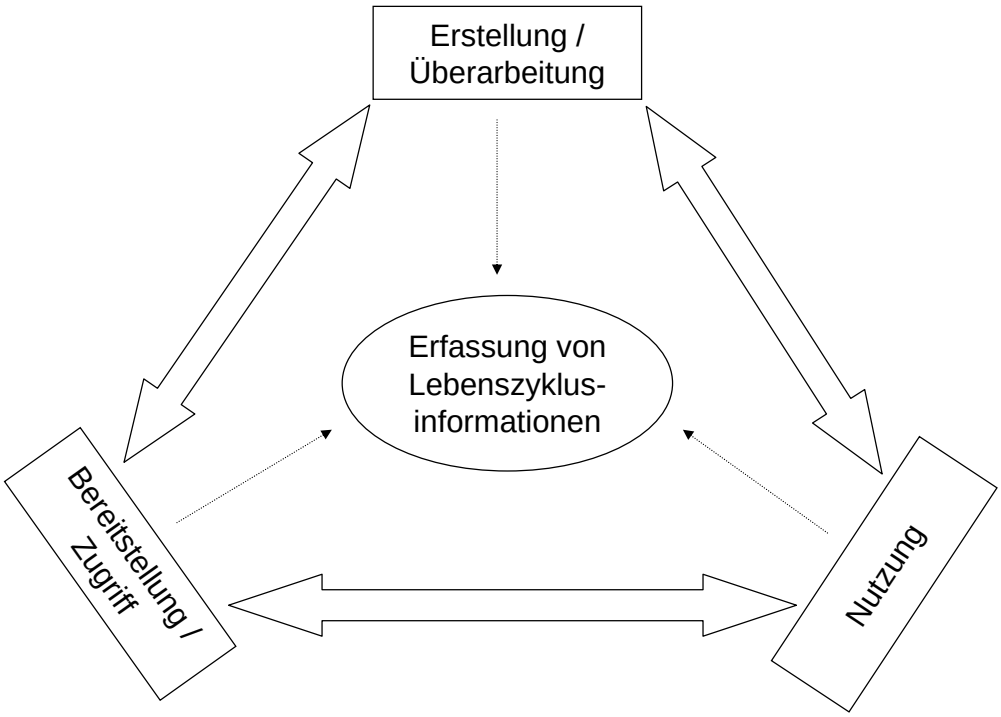

Abbildung 2.8.: Lebenszyklusmodell für Wissendokumente

Es gibt diverse Prozesse, die die Entstehung eines neuen Wissensdokuments mit eigenem Lebenszyklus zur Folge haben. Der Lebenszyklus ist hierbei an die physikalische Instanz des Dokuments gebunden. Wird ein Wissensdokument vervielfältigt, so entstehen neue Wissendokumente mit neuen Lebenszyklen. Wichtig ist es hierbei jedoch, die Verbindungen, die zwischen den physikalischen Instanzen eines Wissensdokuments entstehen, zu erfassen und zu erhalten (siehe Kapitel 2.4.3). So können Informationen, welche für unterschiedliche Instanzen eines Dokuments erfasst wurden, zusammengeführt werden.

Um die Prozesse, die im Lebenszyklus eines Wissendokuments ablaufen, identifizieren zu können, ist es notwendig, die Systeme und Applikationen, die an diesem Lebenszyklus beteiligt sind, zu kennen. Im

Folgenden werden daher jeder Phase des entworfenen Lebenszyklusmodells die entsprechenden Applikationen und Prozesse zugeordnet.

Bereitstellung / Zugriff: Bereitstellung und Zugriff findet üblicherweise innerhalb derselben Umgebung statt. Im Fall von Wissensdokumenten sind dies meistens *Dateisysteme*. Es kann sich, je nach Anwendungsszenario, jedoch auch um *Dokumentenmanagementsysteme (DMS), Digitale Bibliotheken, Contentmanagementsysteme oder Repositories* handeln. In speziellen Fällen kann es vorkommen, dass speziell für den Zugriff eigene Applikationen benutzt werden, wie beispielsweise *Suchmaschinen*. Da die meisten Systeme zur Bereitstellung von Wissensdokumenten die grundlegenden Funktionalitäten dieser Applikationen auch enthalten, ist keine gesonderte Betrachtung notwendig. Die Prozesse, die in dieser Phase des Lebenszyklus ablaufen können, sind sehr heterogen, da es hier die größten Unterschiede zwischen den verwendbaren Applikationen gibt. Zu den Prozessen, die während dieser Phase ablaufen, zählen:

- *Suchen*: Ein Dokument wird über eine Suchmaske gesucht. Meistens geschieht das mit Hilfe von Schlagworten.
- *Browsen*: Alternativ oder ergänzend zur Suche kann ein Dokument auch durch Browsen, zum Beispiel in einem Dateisystem oder Katalog, aufgefunden werden.
- *Selektieren*: Zur Selektion zählt die Auswahl eines bestimmten Suchergebnisses, ein Herunterladen des Dokuments (sofern notwendig), aber auch das Betrachten der Vorschau, beispielsweise in Form eines Thumbnails.
- *Einordnen*: Hierbei wird ein Dokument in eine bestimmte Struktur eingeordnet. Dies kann zum Beispiel durch Speichern in einem bestimmten Ordner oder durch Kategorisierung in einem Katalog erfolgen.

Erstellung / Überarbeitung: Wie in Kapitel 2.2.2 erläutert, gibt es im E-Learning-Bereich oftmals spezielle Werkzeuge zur Unterstützung der verschiedenen Autorenprozesse. Dies ist bei Wissensdokumenten nicht der Fall. Hier wird in den häufigsten Fällen für die Erstellung und Überarbeitung ein und dieselbe Applikation verwendet. Oftmals handelt es sich hierbei um Applikationen, die auf eine bestimmte Art von Dokumenten spezialisierte sind, und die ein bestimmtes Format unterstützen. In solchen Werkzeugen laufen in dieser Phase des Lebenszyklus die in Abschnitt 2.2.2 beschriebenen Erstellungs- oder Überarbeitungsprozesse ab.

Nutzung: Oft erfolgt die Nutzung von Wissensdokumenten in denselben Applikationen, in denen auch die Erstellung und Überarbeitung erfolgt. In vielen Fällen gibt es jedoch auch spezielle Anwendungen, die nur für die Nutzung eines Dokuments gedacht sind. Zudem unterscheiden sich die Nutzungsprozesse von Erstellungs- und Überarbeitungsprozessen so stark, dass eine Unterscheidung sinnvoll ist. Wissensdokumente können je nach Typ auf sehr unterschiedliche Weise genutzt werden, Präsentationen werden gewöhnlich vorgeführt, Textdokumente gelesen, andere wiederum mit einer speziellen Software abgespielt.

Erfassung: Für die Erfassung existieren noch keine dedizierten Applikationen. Zwar gibt es vereinzelt Applikationen, die Lebenszyklusinformationen der Dokumente, die mit ihnen verarbeitet werden, erfassen (siehe Kapitel 3), jedoch erfolgt dies weder in geeignetem Umfang und Format, noch ist für eine Übertragbarkeit der Informationen auf andere am Lebenszyklus beteiligte Systeme gesorgt. Ziel dieser Arbeit ist es, unter anderem diesen Umstand zu ändern.

2.4 Lebenszyklusinformationen und Metadaten

Im folgenden Abschnitt erfolgt die Definition des Begriffs 'Lebenszyklusinformation' und dessen Positionierung zum Begriff 'Metadaten'. Es wird ein hierarchisches Modell zur Einordnung von Lebenszyklusinformationen vorgestellt, welches diese in Verwendungs- und Beziehungsinformationen und weitere Subtypen aufteilt. Anhand des entworfenen Lebenszyklus wird schließlich analysiert, welche Arten von Lebenszyklusinformationen in welchen Phasen des Lebenszyklus auftreten.

2.4.1 Definition

Der Begriff 'Lebenszyklusinformationen' wird im Folgenden aus dem Begriff 'Metadaten' abgeleitet.

Metadaten werden als Daten über andere Daten bezeichnet. Oft wird der Begriff 'Metadaten' mit bestimmten Anforderungen verknüpft. Eine verbreitete Anforderung ist, dass Metadaten sowohl von Autoren als auch rechnergestützt verarbeitet und genutzt werden können [IEE02].

Es gibt unterschiedliche Möglichkeiten, Metadaten zu kategorisieren. Gilliland [Gil08] wie auch Maier [MP05] teilen Metadaten in drei Kategorien ein. Dabei werden zusätzlich zwischen dem beschriebenen Informationsobjekt intrinsischen und extrinsischen Metadaten unterschieden. Intrinsische Metadaten werden dem Objekt bei seiner Entstehung mit auf den Weg gegeben werden. Extrinisch sind Metadaten, die dem Informationsobjekt zu einem späteren Zeitpunkt, oftmals nicht durch den Erzeuger des Objektes selbst, angehängt werden. Folgende Kategorien werden unterschieden:

- *Inhaltsmetadaten* beziehen sich auf den Inhalt des beschriebenen Objekts oder darauf, um was es in einem Dokument geht. Dieser Typ Metadaten ist intrinsisch.
- *Kontextmetadaten* bilden Aspekte ab, die mit der Erstellung oder Nutzung eines Objekts zusammenhängen (wo, was, wann, warum und wie-Aspekte). Diese Metadaten sind extrinsisch.
- *Strukturmetadaten* beziehen sich auf Verbindungen innerhalb von Objekten oder zwischen unterschiedlichen Objekten. Maier zählt auch technische Metadaten wie Dateiformat oder verwendete Kompressionsalgorithmen zu Strukturmetadaten. Diese Form der Metadaten kann intrinsisch, extrinsisch oder beides sein.

Diese Kategorisierung zeigt, dass Informationen über den Entstehungs- oder Nutzungsprozess von Informationsobjekten, also die aufgeführten Kontextmetadaten, ein wichtiger Bestandteil der Metadaten sind. Bei dieser Form der Kategorisierung steht jedoch die Frage im Vordergrund, was die Metadaten beschreiben, also Inhalt, Entstehungs- bzw. Nutzungskontext oder Struktur eines Objekts. Im Rahmen dieser Dissertation ist jedoch die Art der Entstehung von Metadaten beziehungsweise die Dauerhaftigkeit ihrer Verfügbarkeit interessant.

Im Rahmen dieser Arbeit werden Metadaten in prozessabhängige und objektabhängige Metadaten unterteilt. Ausschlaggebend für die Einteilung ist hierbei wie beschrieben die Art und Weise der Entstehung und die Dauerhaftigkeit ihrer Verfügbarkeit. Objektabhängige Metadaten sind an ein bestimmtes Objekt gebunden und spiegeln dessen Eigenschaften wider, existieren also mit dem Objekt oder können anhand des Objektes erzeugt werden. Hierzu zählt beispielsweise das Dateiformat eines Dokuments, aber auch

inhaltliche Metadaten wie Beschreibung, Titel oder Stichworte. Auch nach der Definition von Gilliland als extrinsisch eingeordnete Metadaten wie Informationen über Lizenzen, Rechte oder Katalogisierungsinformationen gehören zu den objektabhängigen Metadaten, da sie der Autor oder ein Experte anhand des vorliegenden Objekts erzeugt. Viele der Felder verbreiteter Metadatenstandards wie LOM [IEE02] oder Dublin Core [DCM08] stellen, dieser Unterteilung folgend, objektabhängige Metadaten dar, da sie anhand des vorliegenden Objekts rekonstruiert oder reproduziert werden können. Dies ist beispielsweise bei der Dateigröße der Fall. Ähnliches gilt für inhaltliche Metadaten wie Beschreibung oder Stichworte. Die Nachvollziehbarkeit setzt nicht notwendigerweise eine Automatisierung voraus, sondern kann und muss teilweise auch durch persönliche Beurteilung erfolgen.

Prozessabhängige Metadaten hingegen sind in ihrer Entstehung an einen bestimmten Prozess gebunden. Ein Beispiel für ein prozessabhängiges Metadatum, welches aus einem Nutzungprozess heraus entsteht, ist die Zeit, die ein Nutzer benötigt, um einen Foliensatz zu präsentieren. Diese Information ist an den Prozess des Präsentierens gebunden. Wenn die Information während dieses Prozesses nicht festgehalten wird, ist sie verloren. Prozessabhängige Metadaten können anhand des Objekts im Nachhinein nicht vollständig rekonstruiert werden. Ausgehend von dieser Grundlage wird der Begriff Lebenszyklusinformation wie folgt definiert:

Definition 2.3 ***Lebenszyklusinformationen*** *sind prozessabhängige Metadaten eines Dokuments. Sie treten in Folge von Prozessen auf, die ein Dokument während seines Lebenszyklus durchläuft. Die Lebenszyklusinformationen eines Dokument können anhand des Dokuments nicht vollständig rekonstruiert werden und sind nur für die Dauer des Prozesses verfügbar.*

Lebenszyklusinformationen sind per Definition dokumentenzentriert. Sie werden also für jedes Dokument aus der Sicht dieses Dokuments gesammelt und nicht aus der Sicht eines Nutzers, Systems oder einer Applikation.

Lebenszyklusinformationen müssen nicht notwendigerweise genau einem bestimmten Dokument zugeordnet werden können. So setzt der Prozess der Wiederverwendung von Dokumenteninhalten die zwei an diesem Prozess beteiligten Dokumente in eine Beziehung. Die Information ist demnach zwei Dokumenten zugeordnet. Dementsprechend kann man Lebenszyklusinformationen weiter unterteilen in *Verwendungsinformationen* und *Beziehungsinformationen*.

Definition 2.4 ***Verwendungsinformationen*** *sind eine Untermenge der Lebenszyklusinformationen. Sie sind immer genau einem Objekt zugeordnet.*

Verwendungsinformationen sind also Informationen, die in Folge bestimmter Prozesse entstehen, die *ein* Objekt im Laufe seines Lebenszyklus durchläuft und die diesem Objekt zugeordnet sind. Sie sind mit den von Gilliland definierten Kontextmetadaten vergleichbar.

Beziehungsinformationen sind wie folgt definiert:

Definition 2.5 ***Beziehungsinformationen*** *sind eine Untermenge der Lebenszyklusinformationen. Durch sie werden zwei Objekte in eine bestimmte Relation gesetzt.*

Beziehungsinformationen können demzufolge die Menge der Beziehungen, die ein Objekt im Laufe seines Lebenszyklus zu anderen Objekten hat, abbilden. Abbildung 2.9 verdeutlicht die beschriebene Unterteilung, die in den folgenden Abschnitten vertieft wird.

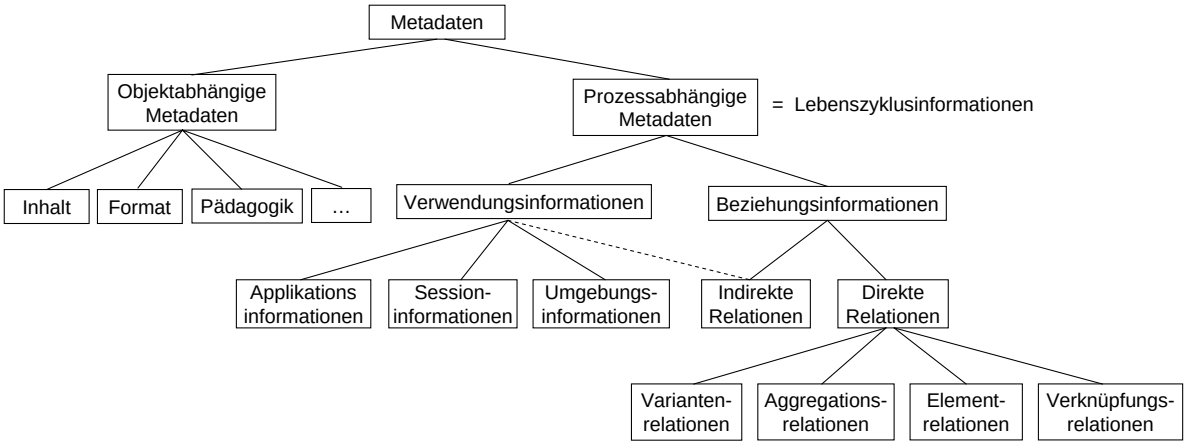

Abbildung 2.9.: Metadaten Taxonomie

Beziehungs- und Verwendungsinformationen werden im Folgenden näher erläutert und an Beispielen beschrieben.

2.4.2 Verwendungsinformationen

Die *Verwendung* eines Dokuments, wie sie in diesem Zusammenhang verstanden wird, findet in allen Phasen des Lebenszyklus statt und ist nicht auf die Nutzungsphase beschränkt. Daher zählen auch Prozesse, die während der Überarbeitungs- und Bereitstellungsphase stattfinden, als Verwendung. Aus Verwendungsprozessen während des Lebenszyklus eines Dokuments entstehen Verwendungsinformationen. Prinzipiell kann jede Information, die aus einem dieser Prozesse entsteht, als Verwendungsinformation betrachtet und erfasst werden. Dabei kann sich eine große Menge an Verwendungsinformationen ergeben. Dies soll beispielhaft anhand des Nutzungsprozesses (*Präsentieren*) einer Präsentation verdeutlicht werden.

1. Die Präsentation wird geöffnet. Hierbei können Zeitpunkt, Nutzer, genutzte Applikation, Computername, ggf. IP-Adresse und - soweit verfügbar - Lokation erfasst werden. Weiterhin kann festgehalten werden, welche anderen Applikationen oder Dokumente zum Zeitpunkt des Öffnens ebenfalls geöffnet waren.

2. Der Foliensatz wird präsentiert. Hierbei kann z.B. erfasst werden, wie viel Zeit der Präsentierende auf jeder Folie verbringt oder die Gesamtdauer der Präsentation. Auch die Applikationen und Prozesse, die während der Präsentation im Betriebssystem gestartet wurden, können als Verwendungsinformationen erfasst werden.

3. Die Präsentation wird geschlossen. Hier kann zum Beispiel ein Zähler, der angibt, wie oft der Foliensatz präsentiert wurde, gesetzt oder der Zeitpunkt des Schließens der Präsentation erfasst werden.

Dieses Beispiel zeigt, dass Verwendungsinformationen sehr heterogen sein können. Sie werden daher in unterschiedliche Kategorien eingeordnet. In dieser Arbeit werden Verwendungsinformationen in *Applikationsinformationen*, *Umgebungsinformationen* und *Sessioninformationen* unterteilt.

Applikationsinformationen entstehen durch explizite Aktionen des Nutzers in der genutzten Applikation. Je nach Aktion und Lebenszyklusphase können sich diese Informationen stark unterscheiden. In der Nutzungsphase zählt zum Beispiel die Dauer der Nutzung dazu - bei der Suche können die eingegebenen Suchbegriffe erfasst werden. Downloads oder Ansichten von Dokumenten erzeugen ebenfalls Verwendungsinformationen dieser Kategorie. Bei der Erstellung und Überarbeitung von Dokumenten gibt es sehr unterschiedliche Aktionen auf beliebiger Granularität ("Dokument geöffnet", "Format geändert", "Textausrichtung geändert", "Wort gelöscht" ...), die zur Entstehung dieser Art von Informationen führen können.

In die Kategorie *Umgebungsinformationen* fallen alle Informationen, die nicht direkt aus der Verwendung eines Dokumentes entstehen, sondern sich aus dem Zusammenhang, in dem das Dokument während seiner Verwendung steht, ergeben. Hierzu zählen beispielsweise Applikationen oder Dokumente, die zum Zeitpunkt der Verwendung des Dokuments ebenfalls geöffnet sind, Dokumente die im selben Ordner gespeichert sind oder mit diesem Dokument in der Ergebnisliste einer Suche stehen. Anhand dieser Informationen ist es auch möglich, verschiedene Dokumente zu gruppieren oder Clustern zuzuweisen.

Zu *Sessioninformationen* zählen Zeitpunkt und Dauer der Verwendung des Dokuments, Benutzer- sowie Computername oder auch IP- oder MAC-Adresse. Durch die Erfassung dieser Informationen können andere Verwendungsinformationen einem bestimmten Nutzer, Zeitpunkt und System zugeordnet werden. Diese Informationen sind bei einer Verwendung grundsätzlich verfügbar.

Sessioninformationen sind in jeder Phase des Lebenszyklus vorhanden. Umgebungsinformationen und Applikationsinformationen sind phasenabhängig. Im Folgenden wird daher der Lebenszyklus daraufhin untersucht, welche Arten von Verwendungsinformationen dieser beiden Kategorien in welchen Phasen auftreten.

Bereitstellung / Zugriff: Umgebungsinformationen treten hier vor allem durch die Struktur der Systems, in dem die Dokumente bereitgestellt werden, auf. Bei einem einfachen Dateisystem können Informationen über Dokumente gesammelt werden, die mit dem aktuellen Dokument im selben Ordner abgelegt sind oder waren. Bei Repositories oder Dokumentenmanagementsystemen kann dies z.B. gleichermaßen mit einem Katalog geschehen. Bei der Suche können Informationen über andere Dokumente im Suchergebnis erfasst werden. Beim Browsing können Dokumente erfasst werden, die kurz vor oder kurz nach dem Ansteuern des entsprechenden Dokuments angesehen wurden.

Applikationsinformationen ergeben sich zum Beispiel aus Suchprozessen in Form der verwendeten Suchbegriffe. Natürlich spielen auch die für Suche oder Browsing verwendeten Applikationen eine Rolle. Außerdem kann jedes Herunterladen und jede Ansicht gezählt und als Verwendungsinformation erfasst werden.

Erstellung / Überarbeitung: In dieser Phase entstehen Umgebungsinformationen, wenn neben dem eigentlichen Erstellungs- und Überarbeitungprozess noch andere Prozesse in anderen Applikationen oder auf anderen Dokumenten ablaufen. Daraus kann ein impliziter Zusammenhang geschlossen werden.

Applikationsinformationen sind in dieser Phase vielfältig. Angefangen beim Öffnen des Dokuments über einzelne Schritte der Erstellungs- und Überarbeitungsprozesse selbst, bis hin zum Schließen des Dokuments kann jede einzelne Aktion ("Folie hinzugefügt", "Bild gelöscht", "Textformat geändert") als Applikationsinformation erfasst werden. Dazu zählen auch Aktionen, die die Entstehung von Wiederver-

Tabelle 2.1.: Verwendungsinformationen in den verschiedenen Phasen des Lebenszyklus

Phase	**Applikations-informationen**	**Umgebungs-informationen**	**Session-informationen**
Bereitstellung / Zugriff	- Verwendete Suchbegriffe - Anzahl Downloads - Anzahl / Dauer Detailansichten - Anzahl Bookmarks - Verwendete Applikation (...)	- Dokumente mit gleichem Speicherort - Dokumente in der gleichen Suchergebnisliste - Zum Zeitpunkt des Zugriffs geöffnete Dokumente / Applikationen (...)	IP-Adresse, Nutzername, Zeitstempel, ...
Erstellung / Überarbeitung	- Verwendete Applikation - Dauer der Verwendung - Nutzeraktionen innerhalb der Applikation	- Geöffnete Applikationen - Im selben Zeitraum geöffnete Dokumente - Gleichzeitig ablaufende Prozesse (...)	IP-Adresse, Nutzername, Zeitstempel, ...
Nutzung	- Art der Nutzung - Verweildauer - Präsentations- / Lesedauer - Dauer pro Folien / Seite (...)	- Geöffnete Applikationen - Gleichzeitig oder in kurzem Abstand geöffnete Dokumente - Gleichzeitig ablaufende Prozesse (...)	IP-Adresse, Nutzername, Zeitstempel, ...

wendungsinformationen zur Folge haben können (siehe Abschnitt 2.4.3), wie zum Beispiel "Folie kopiert" oder "Folie eingefügt".

Nutzung: Umgebungsinformationen entstehen hier, ebenso wie in der Erstellungs- und Überarbeitungsphase, durch weitere, während der eigentlichen Nutzung des Dokuments laufende Prozesse oder Applikationen.

Applikationsinformationen sind von der Art der Nutzung abhängig. Oft ist die Dauer der Nutzung, also Präsentations- oder Verweildauer, interessant. Auch die verwendete Applikation und die Art und Weise, wie ein Dokument genutzt wird, zählen zu den Nutzungsinformationen.

Tabelle 2.1 fasst die Kategorien und Beispiele von Verwendungsinformationen anhand der Phasen des Lebenszyklusmodells zusammen.

2.4.3 Beziehungsinformationen

Im Folgenden wird der Begriff 'Beziehungsinformationen' näher erläutert. Hierfür wird zunächst die Relation als solche definiert. Es erfolgt eine Unterscheidung verschiedener Relationstypen sowie eine Analyse des Lebenszyklus daraufhin, welche Relationen in welchen Phasen entstehen können.

Eine Relation, wie sie in dieser Arbeit verstanden wird, ist folgendermaßen definiert:

Definition 2.6 *Eine **Relation** setzt ein Quellobjekt in eine typisierte Beziehung mit einem Zielobjekt.*

Bei den Objekten, die mit Relationen verknüpft werden, handelt es sich um Dokumente oder Teile von Dokumenten. Der Relationstyp ist von dem Prozess abhängig, aus dem die Relation entstanden ist. Je-

de Relation besitzt eine Inverse. Der inverse Relationstyp zur Relation "istTeilVon" lautet beispielsweise "enthält". Die hier betrachteten Relationen treffen eine Aussage über die strukturelle Beziehung zweier Objekte. Ein Beispiel für eine solche Relation ist: "Objekt A istTeilVon Objekt B". Aus jedem Wissensdokument lassen sich strukturelle Relationen ableiten. So kann durch Dekomposition ein Wissendokument in seine Komponenten zerlegt und diese wiederum in einen strukturellen Zusammenhang gebracht werden - zum Beispiel über die Hierarchie ("istTeilVon") oder die Ordnung ("istVorgängerVon") der Komponenten. Viele Wissensdokumente haben ein Objektmodell, welches strukturelle Relationen implizit beinhaltet. Die Folien einer Präsentation haben zum Beispiel stets eine "istTeilVon"-Relation zu der Präsentation, die sie beinhaltet. Des Weiteren hat eine Folie eine "istVorgängerVon"-Relation zu der Folie, die ihr in der Präsentation nachfolgt. Eine Relation kann also sowohl Objekte auf derselben Aggregationsebene verbinden, als auch Objekte, die auf unterschiedlichen Ebenen liegen. Eine "istTeilVon"-Relation setzt immer Objekte auf unterschiedlicher Aggregationsebene in Beziehung, während eine Ordnungsbeziehung ("istVorgängerVon") stets Objekte auf demselben Aggregationslevel verbindet. Je nach Situation kann sich der Aggregationslevel eines identischen Dokuments unterscheiden. So ist ein Dokument, welches ganz und als einzelnes Element in ein anderes eingefügt wurde, auf einem geringeren Aggregationsniveau als das Dokument, in das es eingefügt wurde. Wird derselbe Vorgang umgekehrt ausgeführt, dreht sich auch das Aggregationsverhältnis der Dokumente um.

Durch die Struktur eines Dokuments vorgegebene Relationen sind nach Definition 2.3 nicht als prozessabhängige Metadaten bzw. Lebenszyklusinformationen anzusehen, da sie grundsätzlich und in vollem Umfang aus dem vorliegenden Dokument rekonstruiert werden können. Von besonderem Interesse in dieser Arbeit sind dagegen Relationen, die nicht implizit durch die vorhandene Struktur eines Dokuments gegeben sind, sondern erst durch bestimmte Prozesse entstehen. Dies geschieht vor allem durch Wiederverwendung. Relationen, die durch Wiederverwendungsprozesse entstehen, werden *Wiederverwendungsrelationen* (oder Wiederverwendungsbeziehungen) genannt. Aus der in Abschnitt 2.2.2 vorgenommenen Kategorisierung der Wiederverwendung von Wissensdokumenten werden folgende Typen von Wiederverwendungsrelationen abgeleitet (vergl. Abbildung 2.9):

- Eine *Aggregationsrelation* ("istTeilVon / enthält") für Libbrechts erste Kategorie (Unveränderte Übernahme)
- Eine *Elementrelation* ("enthältElementAus / liefertElementAn") für zweite Kategorie (Kopieren und Einfügen)
- Eine *Variantenrelation* ("istVarianteVon / hatVariante") für Libbrechts dritte Kategorie (Kopieren und Überarbeiten)
- Eine *Verknüpfungsrelation* ("verlinktAuf / wirdVerlinktVon") für Libbrechts vierte Kategorie (Verknüpfung)

Abbildung 2.10 verdeutlicht die unterschiedlichen Relationstypen. Eine Elementrelation verbindet zwar zwei Dokumente auf einem hohen Aggregationslevel, die wiederverwendeten Elemente innerhalb der Dokumente sind jedoch auch von Interesse. Deshalb sollten bei dieser Relation zusätzliche Informationen über diese Elemente festgehalten werden. Der Begriff Variante abstrahiert von vielfältigen Beziehungen, die zwischen zwei durch eine solche Relation verbundenen Dokumenten bestehen können. Eine

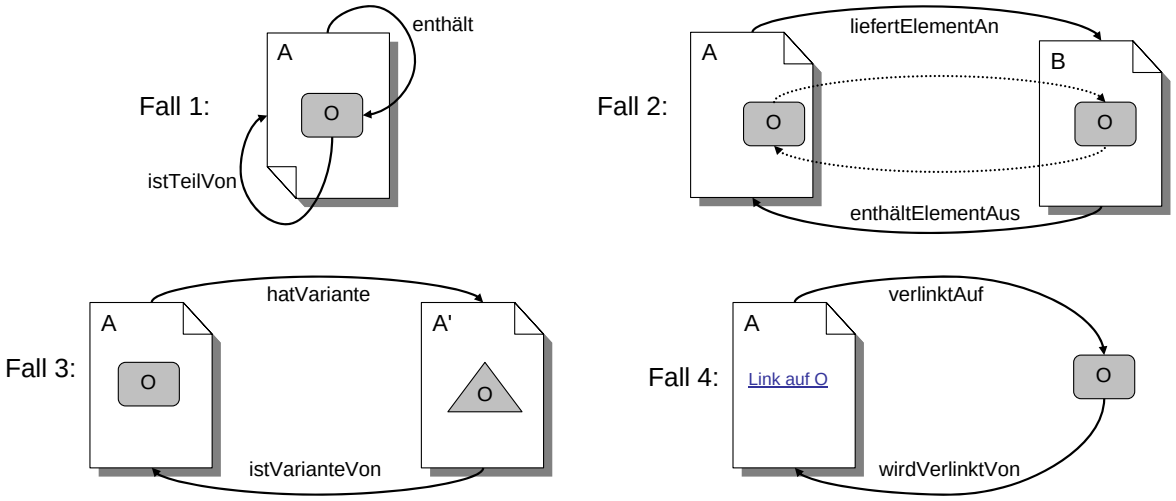

Abbildung 2.10.: Typen von Wiederverwendungsrelationen

Variantenbeziehung, wie sie in dieser Arbeit verwendet wird, umfasst daher mehr als den Begriff der Variante, wie er üblicherweise bei der Versionierung gebräuchlich ist. Varianten sind dort als Abzweigung eines Versionsstrangs zu sehen, während Versionen selbst die Entwicklung eines Dokuments über die Zeit darstellen [Lie01]. In dieser Arbeit kann eine Variante beide dieser Fälle umfassen. So kann es sich bei Varianten um identische Kopien handeln, eine Variante kann leicht abgeändert und aktualisiert werden oder anhand der in [ZRS06] beschriebenen Prozesse in Layout, Format, Zielgruppe oder Sprache angepasst werden. Im E-Learning-Bereich gibt es hierfür spezielle Werkzeuge, die diese Prozesse unterstützen. Dadurch ist es möglich, spezielle Typen von Variantenbeziehungen zu erfassen, die der Relation teilweise eine Semantik zuweisen (siehe [LHRS07]). Dies ist bei Wissensdokumenten nicht der Fall, da diese expliziten Anpassungsprozesse von den für Wissensdokumente verwendeten Werkzeugen nicht besonders unterstützt werden, so dass eine Erfassung des speziellen Typs einer Variantenrelation nicht möglich ist. Wiederverwendungsrelationen können zwar teilweise rekonstruiert werden, dies setzt jedoch voraus, dass zu diesem Zeitpunkt sowohl Quell- als auch Zielobjekt verfügbar sind. Daraufhin können durch Analyse der Dokumente (manuell oder mit Hilfe von Post-Processing-Techniken) Rückschlüsse auf eine Wiederverwendungsrelation gezogen werden. Zudem kann hierdurch weder der Erzeuger der Relation, noch der Zeitpunkt, noch die Orientierung der Relation auf diese Weise bestimmt werden. Insbesondere Letzteres ist aber eine für viele Anwendungsfälle wichtige Information (siehe Kapitel 4.3). Deshalb werden Wiederverwendungsbeziehungen hier als prozessabhängig angesehen.

Neben den Wiederverwendungsbeziehungen und den Strukturrelationen gibt es auch noch Beziehungen, die aus Verwendungsinformationen gefolgert werden können. Es stehen zum Beispiel alle Dokumente einer Ergebnisliste derselben Suchanfrage in einer bestimmten Beziehung zueinander oder, wie im Beispielszenario vom Anfang des Kapitels gegeben, die Vorträge derselben Session auf einer Konferenz. Auch Dokumente, auf die kurz nacheinander zugegriffen wird, verbindet eine Relation, die aus den entsprechenden Verwendungsinformationen geschlossen werden kann. Diese Relationen werden als *indirekte Relationen* bezeichnet und sind sowohl Beziehungs- als auch Verwendungsinformationen zugeordnet (vergl. Abbildung 2.9).

Tabelle 2.2.: Beziehungsinformationen in den Phasen des Lebenszyklus

Phase	**Direkte Relationen (Wiederverwendungsbeziehungen)**	**Indirekte Relationen**
Bereitstellung / Zugriff	- Variantenrelationen	- Gleicher Speicherort (z.B. Ordner) - Gleiche Suchergebnisliste (...)
Erstellung / Überarbeitung	- Elementrelationen - Aggregationsrelationen - Variantenrelationen - Verknüpfungsrelationen	- Kurz nacheinander geöffnete Dokumente
Nutzung	- keine direkten Relationen	- Dokumente mit gleichem Nutzungszeitpunkt, -ort und -zweck

Im Folgenden werden die Phasen des Lebenszyklus auf Prozesse hin untersucht, die zur Entstehung von Beziehungsinformationen führen (siehe Abbildung 2.2).

Bereitstellung / Zugriff: In dieser Phase können Beziehungsinformationen bei der Suche oder der Einordnung entstehen. So stehen z.B. alle Dokumente in Beziehung zueinander, die gemeinsam in der Ergebnisliste einer bestimmten Suchanfrage waren. Beim Einordnen in ein Dateisystem können Relationen zwischen Dokumenten im selben Ordner entstehen. Auch können Dokumente in Relation gesetzt werden, auf die innerhalb einer Sitzung zugegriffen wurde (zum Beispiel durch Herunterladen aus einem Repository). All diese Informationen können jedoch entweder rekonstruiert oder aus den entsprechenden Verwendungsinformationen abgeleitet werden. Hierbei ist die Relation nicht Ergebnis eines bestimmten Prozesses, sondern kann aus dem Ergebnis des Prozesses bzw. aus den Informationen, die während des Prozesses entstehen, abgeleitet werden. Diese indirekten Relationen werden daher nicht nur zu Beziehungsinformationen gezählt, sondern auch zu Verwendungsinformationen, da sie aus diesen abgeleitet werden können.

Nutzung: Während der Nutzung eines Dokuments selbst entstehen keine Beziehungsinformationen. Es können jedoch indirekte Beziehungen aus gesammelten Verwendungsinformationen abgeleitet werden. So kann unter Umständen eine Relation zwischen Dokumenten aus der Tatsache abgeleitet werden, dass zwei Dokumente zum selben Zeitpunkt oder in kurzem zeitlichen Abstand genutzt oder betrachtet wurden. So sind die verschiedenen, zeitlich eng aufeinanderfolgenden Vorträge innerhalb einer Sitzung bei einer wissenschaftlichen Konferenz häufig inhaltlich verwandt. Auch hierbei handelt es sich um indirekte Relationen.

Erstellung / Überarbeitung: Diese Phase ist die Hauptphase, was die Entstehung von Relationen betrifft. Bei Durchführung von Wiederverwendungs- und Überarbeitungsprozessen entstehen explizite Wiederverwendungrelationen. Die Informationen, die durch einen solchen Prozess entstehen, unterscheiden sich, abhängig von der verwendeten Applikation. Ein Datenmodell zur Verwaltung der Informationen sollte dies berücksichtigen.

2.5 Zusammenfassung

In dem vorliegenden Kapitel erfolgte eine tiefergehende Motivation des Ansatzes zur Erfassung von Lebenszyklusinformationen anhand eines repräsentativen Beispielszenarios. Wissensdokumente als Gegenstand der Arbeit wurden definiert und von den verwandten Lernobjekten und Lernressourcen auch im Bezug auf verschiedene Autorenprozesse abgegrenzt. Der in dieser Arbeit wichtige Begriff der Wiederverwendung wurde definiert und auf Wissensdokumente übertragen. Aufbauend auf vorhandenen Modellen wurde ein Lebenszyklusmodell für Wissensdokumente entwickelt. Auf dieser Basis wurden die in den verschiedenen Phasen des Modells anfallenden Informationen identifiziert und kategorisiert. Es wurde gezeigt, dass in den verschiedenen Phasen des Lebenszyklus' eines Wissensdokuments eine Vielzahl verschiedener Informationen entstehen. Da Applikationen, mit denen Wissensdokumente normalerweise verarbeitet werden, diese jedoch nicht erfassen, besteht die Gefahr, dass die Informationen verloren gehen.

Um dies zu verhindern und so Lebenszyklusinformationen nutzbar zu machen, muss ein System folgende Kriterien erfüllen:

1. Die Informationen müssen dort erfasst werden, wo sie entstehen - also innerhalb der entsprechenden Applikationen bzw. Systeme.

2. Wiederverwendung als wichtiger Aspekt im Lebenszyklus von Wissensdokumenten muss berücksichtigt werden - zum Beispiel durch Erfassung von Beziehungsinformationen.

3. Die Informationen müssen systemübergreifend bereitgestellt werden, also nach der Erfassung anderen Systemen zur Verfügung gestellt werden. Es muss gewährleistet sein, dass die Informationen die Grenzen zwischen den verschiedenen Applikationen und Systemen, wo sie entstehen und erfasst werden, überwinden.

4. Damit die Informationen genutzt werden können, müssen sie verarbeitet und in ein maschinenlesbares Format überführt werden.

5. Die Erfassung sollte den gesamten Lebenszyklus eines Dokumentes abdecken.

6. Die erfassten Informationen sollten verlässlich sein.

Im folgenden Kapitel werden zunächst existierende Ansätze zur Erfassung, Verwaltung und Nutzung solcher Informationen untersucht, verglichen und anhand der genannten Kriterien beurteilt.

Teil II.

Erfassung, Verwaltung und Nutzung von Lebenszyklusinformationen

3 Verwandte Arbeiten

Dieses Kapitel befasst sich mit verwandten Arbeiten aus dem Bereich der Erfassung, Verwaltung und Nutzung der Lebenszyklusinformationen von Wissensdokumenten und Lernressourcen. Es erfolgt zunächst die Beschreibung von Ansätzen, die sich mit der Erfassung von Lebenszyklusinformationen im weiteren Sinne befassen. Die Erfassung geht dabei zumeist mit einer Verwaltung der Informationen einher. Im zweiten Teil des Kapitels werden Ansätze beschrieben, die sich mit der Nutzung erfasster Lebenszyklusinformationen in diversen Szenarien beschäftigen. Nach Diskussion, Vergleich und Beurteilung der Ansätze werden schließlich die von den existierenden Ansätzen und Systemen offengelassenen Fragestellungen und Herausforderungen identifiziert.

3.1 Ansätze für die Erfassung und Verwaltung von Lebenszyklusinformationen

Die verwandten Ansätze im Bereich der Erfassung von Lebenszyklusinformationen sind im folgenden Abschnitt nach der Erfassung von Verwendungsinformationen und der Erfassung von Beziehungsinformationen unterteilt. Ansätze, die indirekte Beziehungsinformationen erfassen, also Relationen aus Verwendungsinformationen ableiten, werden hierbei der Erfassung von Beziehungsinformationen zugeordnet.

3.1.1 Verwendungsinformationen

Ecological Approach - Brooks & McCalla

In [McC04] schlägt Gord McCalla erstmals die Erfassung von Verwendungsinformationen im E-Learning-Bereich vor. Seinem Vorschlag zufolge sollen Metadaten nicht nur in expliziten Auszeichnungsphasen, sondern auch während der Nutzung von Lernressourcen erfasst werden. Diese Ansatz wird "Ecological Approach" (ökologischer Ansatz) genannt. Er sieht vor, dass Interaktionen zwischen Lernenden und Lernressourcen - dazu zählen Verweilzeit, Zugriffsmuster, Tastatureingaben, Feedback des Lernenden, aber auch technische Aspekte wie Hardware-Konfiguration und verwendete Applikationen - erfasst und genutzt werden. Dabei wird unter Metadaten "der Prozess des logischen Folgerns auf gesammelten Interaktionen von Nutzern mit Lernobjekten" verstanden [BM06]. Die gesammelten Informationen werden genutzt, um Lernermodelle zu erstellen. Diese Modelle werden dann an ein Lernobjekt - also das Dokument - angehängt und mit anderen Lernermodellen an anderen Lernobjekten verbunden. Mit der Zeit wächst die Zahl der Lernermodelle an den Lernobjekten - weshalb der Ansatz "ökologisch" genannt wird. Durch die erfassten Informationen werden die Lernressourcen zu Agenten, welche Informationen über die Lerner, die mit ihnen interagiert haben, verwalten und verarbeiten und sich entsprechend anpassen können. Die gesammelten Lernermodelle können dann nach Mustern durchsucht und für verschiedene Zwecke genutzt werden. Dazu zählen Lernhilfen, Planungshilfen für Lernpfade durch Lernobjekte oder

auch Vorschlagsysteme, die anhand der gesammelten Daten auf den Nutzer zugeschnittene Vorschläge generieren.

Dieser Ansatz ist ein Vorreiter für die Erfassung prozessabhängiger Metadaten. Die Erfassung der Informationen in diesem Ansatz ist nutzerzentriert, allerdings werden die Informationen aus der Sicht eines Dokuments gesammelt: Ein Dokument besitzt also jeweils ein Modell für jeden Nutzer, der mit ihm interagiert hat. Es werden ausschließlich Verwendungsinformationen erfasst und das wiederum lediglich in der Nutzungsphase des Lebenszyklus. Der Fokus liegt bei diesem Ansatz auf webbasierten Dokumenten.

Das CAM-Framework

Das CAM-Framework (Contextualized Attention Metadata) ist ein System zur Erfassung und Verwaltung von Metadaten, die durch die Interaktion eines Nutzers mit digitalen Dokumenten entstehen. Dazu werden verschiedenste Applikationen mit Hilfe von Plug-ins beobachtet. Zu den Applikationen, die vom CAM-System unterstützt werden zählen: Microsoft Office Applikationen, Webbrowser, Mail-Programme, Winamp (Software zum Abspielen von Musik) sowie der MSN Messenger (ein Instant-Messaging-Programm). Interaktionen innerhalb dieser Applikationen werden dann als sogenannter "Attention Stream" an eine zentrale Instanz - den "Attention Store" - gesendet und dort verarbeitet (siehe Abbildung 3.1). Eine "Kontextualisierung" der gesammelten Daten erfolgt dadurch, dass die gesammelten Daten verschiedener Applikationen vereint werden können. So ist es möglich, herauszufinden, welche Prozesse und Aktionen ein Nutzer simultan oder innerhalb einer kurzen Zeitspanne durchgeführt hat. Dies entspricht einer Erfassung von Umgebungsinformationen (Kapitel 2.4.2).

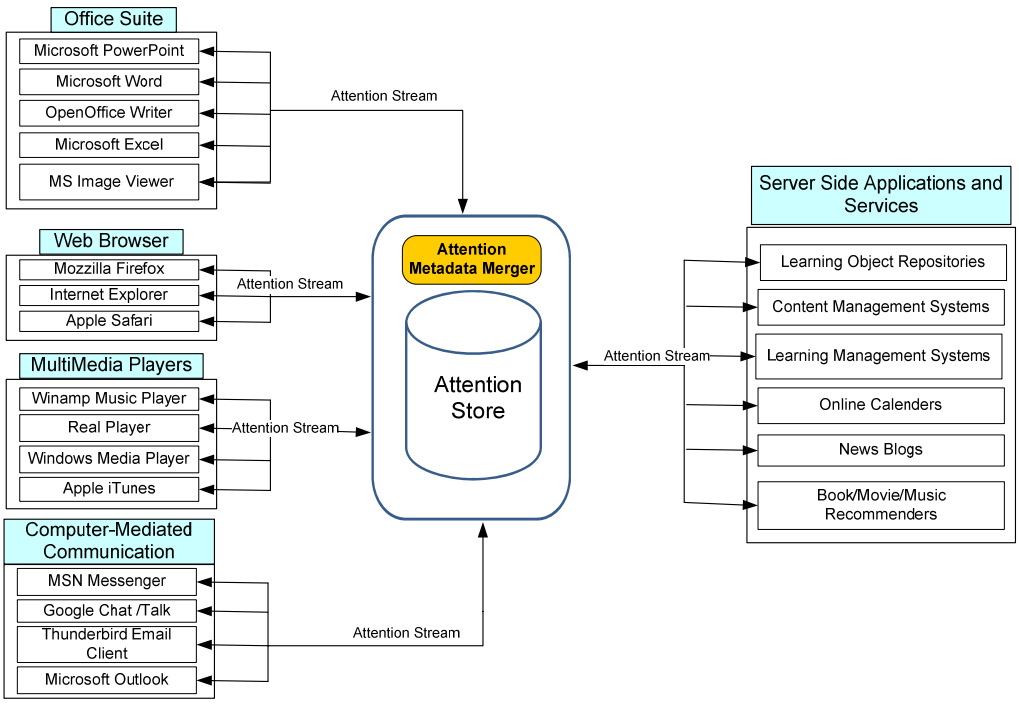

Abbildung 3.1.: Das CAM-Framework [WNVD07]

Die Informationen werden ausschließlich auf Applikationsebene gesammelt und nicht mit Hilfe von Low-Level-Ereignissen wie Tastatur- oder Mauseingaben. Um die gesammelten Informationen vorzuhalten, wurde ein Schema entwickelt, das Contextualized Attention Metadata-Schema. Seinen Ursprung hat es in dem sogenannten attentionXML Format [NMD05], welches in mehreren Iteration weiterentwickelt wurde ([NWD06a, NWD06b]). Hier wird die letzte veröffentlichte Version beschrieben ([WNVD07]).

Das "Group"-Element vereint die Informationen über Interaktionen eines Nutzers in verschiedenen Applikationen mit verschiedenen Dokumenten. Für jede Applikation gibt es innerhalb des Group-Elements einen sogenannten "Feed", der die in dieser Applikation gesammelten Informationen kapselt. Ein Feed wiederum besteht aus diversen "Items", die die digitalen Dokumente repräsentieren. Jedes Dokument kann verschiedenen Arten von Interaktion unterzogen werden. Dies wird mit dem "Event"-Element unterhalb des Item-Elements modelliert. Ein solches Ereignis besteht wiederum aus Informationen über die Interaktion selbst und Informationen über die "Session" (Sessioninformationen siehe Kapitel 2.4.2). Ein Ereignis kann weiterhin beschrieben werden durch ein "Entry"-Element. Hier kann ein eigenes Schema zur Beschreibung des Ereignisses hinterlegt werden. Abbildung 3.2 zeigt das komplette CAM-Schema in Entity-Relationship-Notation wie in [WNVD07] veröffentlicht.

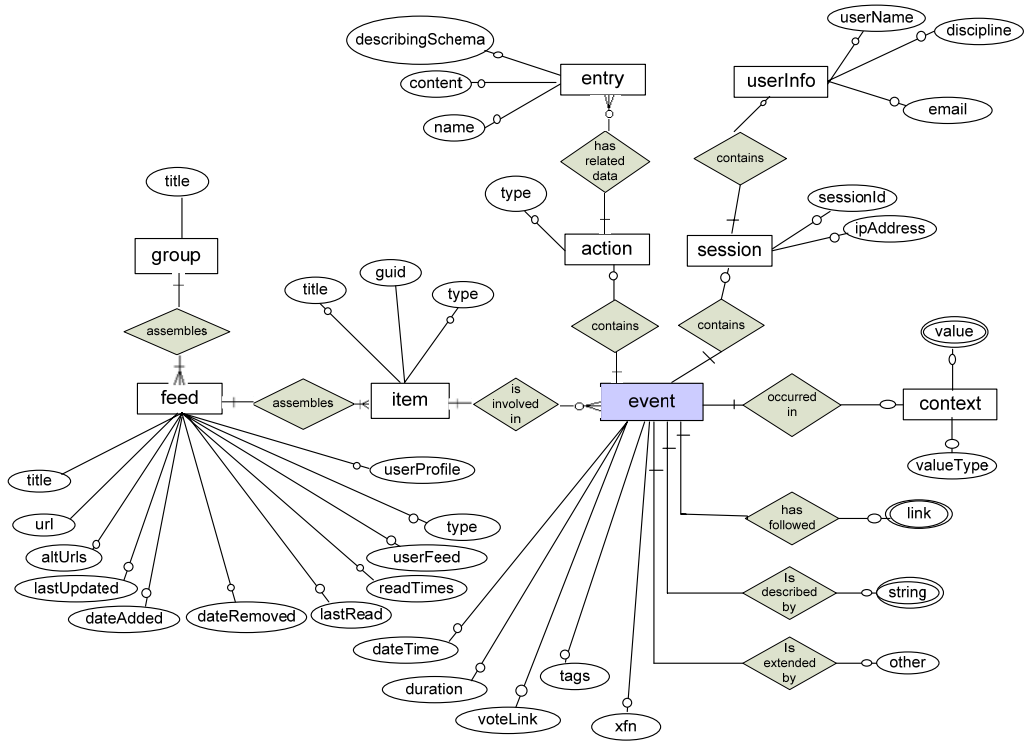

Abbildung 3.2.: Das Contextualised Attention Metadata Schema [WNVD07]

Das CAM-Framework erfasst und verwaltet Verwendungsinformationen auf Applikationsebene. Die Erfassung geschieht aus Sicht eines Nutzers - ein Feed ist immer einem Nutzer zugeordnet. Eine Ausgabe aller zu einem Dokument gesammelten Informationen ist nur eingeschränkt möglich. Während bei zentralisiert gespeicherten Dokumenten, wie Webseiten oder in zentralisierten Systemen vorgehaltene Dokumente, alle Aktionen und Ereignisse auf diese Art und Weise dokumentenzentriert ausgegeben werden können, ist dies bei lokalen Dokumenten, die in verschiedenen Instanzen vorliegen und mit denen

unterschiedliche Nutzer verteilt interagieren, nur eingeschränkt möglich. Hier lassen die Autoren die Frage offen, wie das CAM-Framework eine Verbindung zwischen den Instanzen herstellt. Dies steht jedoch nicht im Fokus des CAM-Systems. Wiederverwendungsinformationen werden vom CAM-Framework ebenfalls nicht berücksichtigt und finden auch keinen Eingang in das entworfene Schema. In Abschnitt 3.2 werden verschiedene Ansätze zur Nutzung von CAM erläutert.

APOSDLE & DYONIPOS

Im Projekt APOSDLE (Advanced Process-Oriented Self-Directed Learning Environment) [MHT+05] ist ein Ziel, Wissensarbeitern kontextsensitive, also situationsabhängige Hilfestellungen zu geben [LG08]. Um dies zu ermöglichen, ist es notwendig, die aktuelle Aufgabe des Wissensarbeiters zu erkennen. Wenn diese bekannt ist, kann der Wissensarbeiter mit auf diese Aufgabe angepassten Dokumenten und Lernressourcen versorgt werden [LFBG07]. Zusätzlich schlägt das System Experten vor, die den Nutzer bei der Bewältigung der Aufgabe unterstützen können [LGF+07]. Um die aktuelle Aufgabe des Wissensarbeiters automatisiert erkennen zu können, werden die Aktionen des Nutzers beobachtet und Informationen über dessen Kontext - also auch Verwendungsinformationen - gesammelt. Hierzu zählen beispielsweise die aktuell geöffneten Dokumente und Applikationen, Tastatureingaben, Änderungen im Dateisystem, in der Zwischenablage oder Mausbewegungen (siehe Abbildung 3.3). Die Auswertung der Informationen erfolgt mit Hilfe von Algorithmen aus dem Bereich des maschinellen Lernens [Mit97]. Hierzu wird das System zunächst durch den Nutzer trainiert, indem dieser dem System mitteilt, wann ein Wechsel der Aufgabe erfolgt ist. Mit Hilfe dieser Information ist das System schließlich in der Lage, Änderungen in der aktuellen Aufgabe eines Wissensarbeiters automatisch zu erkennen.

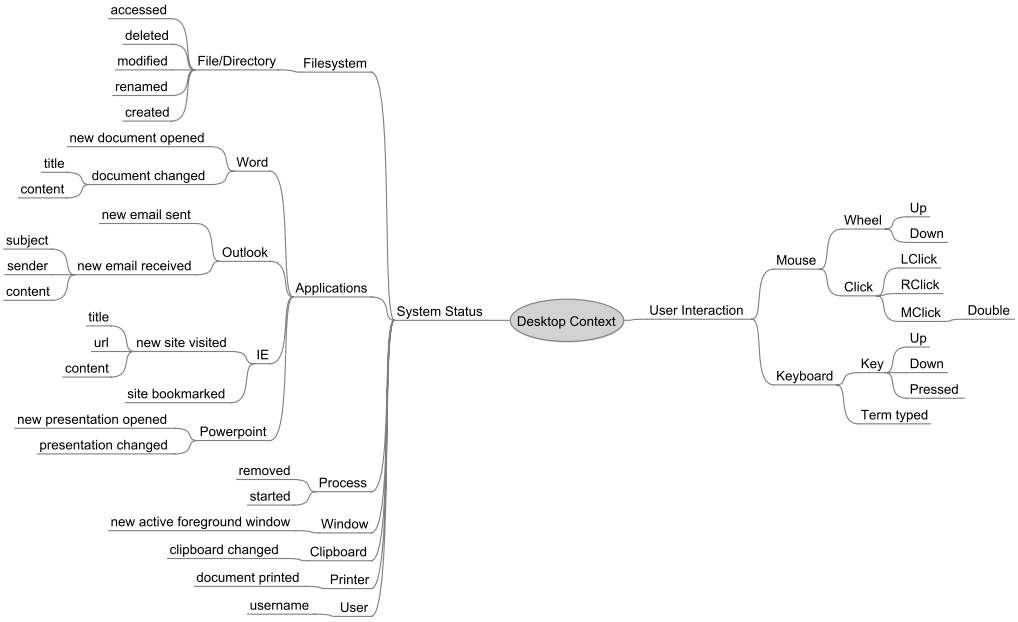

Abbildung 3.3.: Sensoren zur Erfassung von Verwendungsinformationen in APOSDLE [LFBG07]

Das Projekt DYONIPOS (DYnamic ONtology based Integrated Process OptimiSation) [MRT07] verfolgt einen ähnlichen Ansatz: Mit Hilfe von Verwendungsinformationen wird die aktuelle Aufgabe und der

Wissensbedarf eines Nutzers herausgefunden, um ihn gezielt in seiner Arbeit unterstützen zu können. Die Informationen werden auf sehr ähnliche Art und Weise unter Verwendung der gleichen Quellen wie in APOSDLE gesammelt [RDL09]. Unterschiede ergeben sich aus der Art und Weise, wie die Informationen verarbeitet werden. So werden die gesammelten Informationen semantisch aufbereitet. Außerdem wird in [RKL+08] auch die Erfassung von Beziehungen zwischen Dokumenten erwähnt, von den Autoren jedoch nicht vertieft.

Beide Projekte benutzen Verwendungsinformationen, um automatisiert den aktuellen Wissensbedarf und die aktuelle Aufgabe eines Wissensarbeiters zu erkennen. Die gewonnenen Informationen werden dann genutzt, um den Nutzer gezielt zu unterstützen. Beziehungsinformationen werden kaum berücksichtigt.

SHAMAN - Erfassung von Kontextinformationen für die Archivierung

Im Projekt SHAMAN (**S**ustaining **H**eritage **A**ccess through **M**ultivalent **A**rchivi**N**g) [Pro09] geht es um die Langzeitarchivierung digitaler Daten [EBJ+09]. Eine Herausforderung bei der Langzeitarchivierung ist, dass die Daten auch nach einem längeren Zeitraum noch benutzbar, wiederverwendbar und verarbeitbar sein müssen. Dies soll auch in einer Umgebung möglich sein, die sich von der Umgebung, in der die digitalen Objekte ursprünglich erstellt wurden, unterscheidet. Um dies zu gewährleisten sollen unter anderem Kontextinformationen über archivierte Objekte gesammelt werden. Diese beinhalten Informationen darüber, wie der Datenstrom dekodiert und der ursprüngliche Inhalt gewonnen werden kann, in welcher Umgebung das digitale Objekt erstellt wurde, welche Aktoren und Ressourcen beteiligt waren sowie Informationen über organisatorische und technische Prozesse, die in Verbindung mit der Produktion, der Archivierung, dem Zugriff und der Wiederverwendung des digitalen Objekts stehen [BKJH10]. Hierzu wurde ein Modell auf Basis des OAIS (Open Archival Information System) [OAI02] -Referenzmodells entwickelt, das dieses erweitert und die Erfassung der genannten Aspekte unterstützt (vergleiche Abbildung 3.4). Das bestehende OAIS Informationsmodell wird um eine Kontext-Schicht erweitert, die Informationen über den Erstellungs- und Wiederverwendungskontext sowie über den Archivierungskontext enthält. Das bestehende Konzept wurde bisher für die Domäne des wissenschaftlichen Publizierens umgesetzt. Die Kontextinformationen, die hierin gesammelt werden, umfassen

1. Domänenspezifische Begriffe aus der gegebenen Domäne des wissenschaftlichen Publizierens wie "Abstract", "Präsentation", "Proceedings",
2. Informationen über die organisatorische Umgebung wie Personen, Rollen und Zugehörigkeiten,
3. Informationen über domänenspezifische Prozesse und deren Umsetzung wie zum Beispiel "Einreichung", "Review" oder "Indexierung".

Um die gewonnenen Kontextdaten zu modellieren, wird ein auf Ontologien basierendes Modell verwendet. Die Informationen, die in SHAMAN erfasst werden, sind teilweise dokumentenzentriert (1,2), teilweise jedoch auch prozesszentriert (3). Die dokumentenzentrierten Informationen lassen sich als Verwendungsinformationen klassifizieren, auch wenn sie ein Dokument, im Vergleich zu CAM, auf einer abstrakteren Ebene beschreiben.

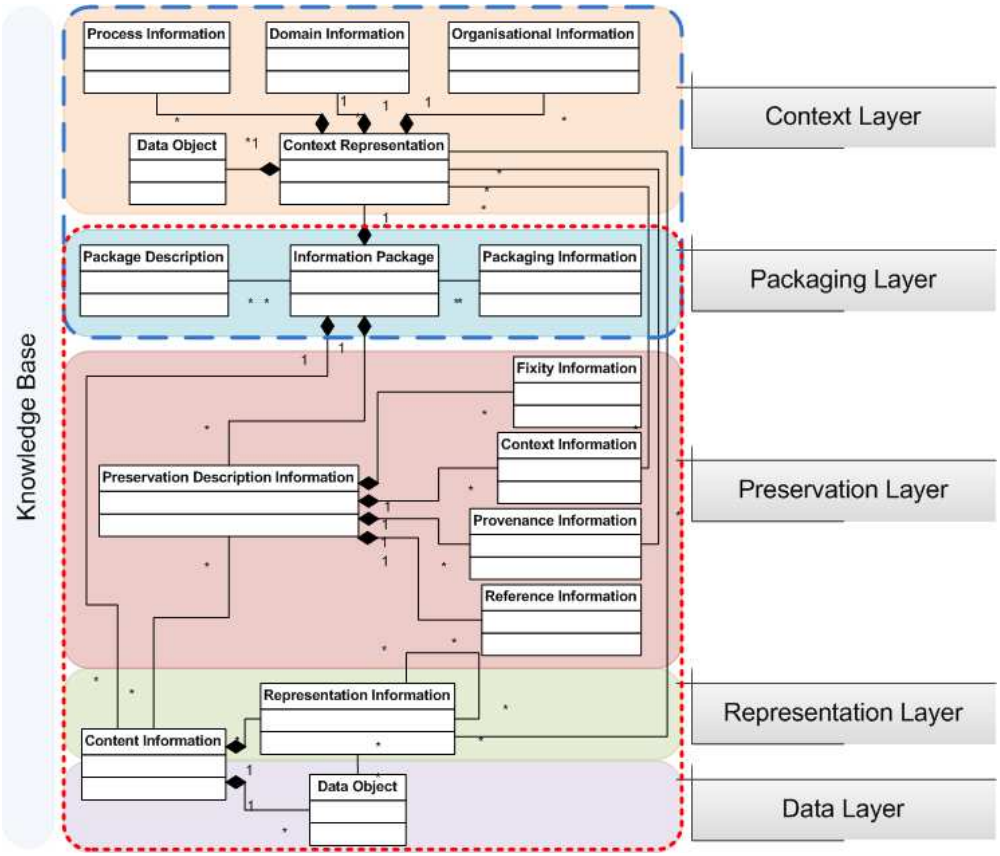

Abbildung 3.4.: Erweiterung von OAIS aus [BKJH10]

Die in SHAMAN erfassten Kontextdaten dienen hautpsächlich dazu, die Wiederverwendbarkeit von archivierten digitalen Objekten zu erhöhen. In [EBJ+09] stellen Engel et al. Möglichkeiten vor, wie die Kontextdaten im Speziellen für das Retrieval digitaler Objekte genutzt werden können. Hierzu zählen die Aufnahme der Kontextdaten in einen zusätzlichen Volltextindex, sowie die Berücksichtigung von Beziehungen zwischen archivierten Objekten. Diese Beziehungen werden über gemeinsame Attribute der digitalen Objekte, wie Autor, Konferenz oder Session hergeleitet, entsprechen also indirekten Relationen.

Kontextsensitive Dienste

Ansätze, wie sie in APOSDLE oder DYONIPOS umgesetzt werden, sehen die von einem Nutzer zu einem bestimmten Zeitpunkt generierten Verwendungsdaten als Kontext dieses Nutzers (siehe [Goe05] für eine ausführliche Diskussion des Kontext-Begriffs). Die zur Erfassung notwendigen Komponenten werden hierbei als Sensoren betrachtet, die den Kontext des Nutzers wahrnehmen. Wenn der Sensor-Begriff weiter gefasst wird und nicht auf reine Software-Sensoren reduziert bleibt, wie es in den beiden genannten Projekten der Fall ist, ergeben sich weitere verwandte Anwendungsbereiche. Auch hier werden Informationen während der Entstehung erfasst, um sie in verschiedenen Szenarien nutzbar zu machen. Einer dieser Anwendungsbereiche ist die Zugriffskontrolle im Ubiquitous Computing. In diesem Bereich ist eine Zugriffskontrolle auf Basis von Passwörtern oder anderen physikalischen Zugriffsmechanismen

unerwünscht und störend. Unter gewissen Voraussetzungen kann eine Zugriffskontrolle anhand von Kontextinformationen geschehen, die von Sensoren automatisiert erfasst werden [Mos02].

Weitere Anwendungen für von physikalischen Sensoren wie Temperatur- oder Bewegungssensoren erfasste Informationen ergeben sich im Bereich der Kommunikationsdienste. Görtz schlägt vor, mit Hilfe von Kontextinformationen, den optimalen Zeitpunkt und das optimale Medium für eine Kommunikation mit einer Person zu ermitteln [GASS04]. Dieser Ansatz wird von Schmitt et al. durch eine Komponente erweitert, die zur Laufzeit durch Feedback der Nutzer lernt und sich so auf die unterschiedlichen Eigenheiten von Nutzern einstellt [SHRS08]. Hierzu werden Technologien aus dem Bereich des Machine-Learning eingesetzt. Das kontextabhängige Anbieten von Diensten wird auch bei Bergstraesser et al. mit Hilfe von erfassten Informationen bewerkstelligt. Die Nutzung der Dienste spielt sich jedoch in den virtuellen Welten von Mehrspieler-Online-Spielen ab. Software-Sensoren, die in die Spiele eingebettet werden können, rufen die aktuelle Situation und die Aktivitäten des Spielers ab und veranlassen, dass dem Spieler, abhängig von den erfassten Daten, verschiedene hilfreiche Dienste angeboten werden ([BHL+07], [BHRS09]).

Es lassen sich noch viele weitere Beispiele und Arbeiten finden, in denen Informationen aus Prozessen gesammelt und auf verschiedene Weise genutzt werden. Die hier genannten sollen einen Überblick über die verschiedenen Möglichkeiten einer solchen Nutzung und die verschiedenen Szenarien, in denen solche Informationen erfasst werden, geben. Sie stehen jedoch nicht in direktem Zusammenhang mit den Beiträgen dieser Arbeit.

3.1.2 Beziehungsinformationen

Es gibt kaum Ansätze, die sich mit der Erfassung von Beziehungsinformationen während des gesamten Lebenszyklus eines Dokuments beschäftigen. Häufig werden jedoch Beziehungen auf andere Art und Weise automatisiert erzeugt oder aus Verwendungsinformationen abgeleitet. Im Folgenden werden die Ansätze beschrieben, welche die größte Verwandtschaft mit dem in dieser Arbeit vorgeschlagenen Ansatz haben.

Das ALOCOM-Framework

ALOCOM [VOD08] ist ein Framework, das zur Unterstützung der Wiederverwendung von Dokumenten konzipiert wurde. Der Name ALOCOM leitet sich aus dem hierfür entworfenen Objektmodell her: **A**bstract **L**earning **O**bject **CO**ntent **M**odel [VGJD05]. Es wurde, wie sich aus dem Namen schließen lässt, ursprünglich für Lernobjekte konzipiert, findet jedoch auch für andere Dokumente wie Folien-Präsentationen Verwendung. Das ALOCOM-Framework ermöglicht die Extraktion modularer Komponenten, also einzelner Teile aus bestehenden Dokumenten, sowie die Wiederverwendung dieser Teile bei der Erstellung neuer Dokumente. Hierzu wurden Plug-ins für Microsoft PowerPoint, Microsoft Word und einen SCORM-kompatiblen Editor für Lernobjekte ("Reload") entwickelt. Die Basis des Systems bildet das ARIADNE-Repository ([ARI09, DFC+01]). Hierin werden die extrahierten Teildokumente über bestimmte Webservice-Schnittstellen [TND+03] gespeichert und wieder abgerufen.

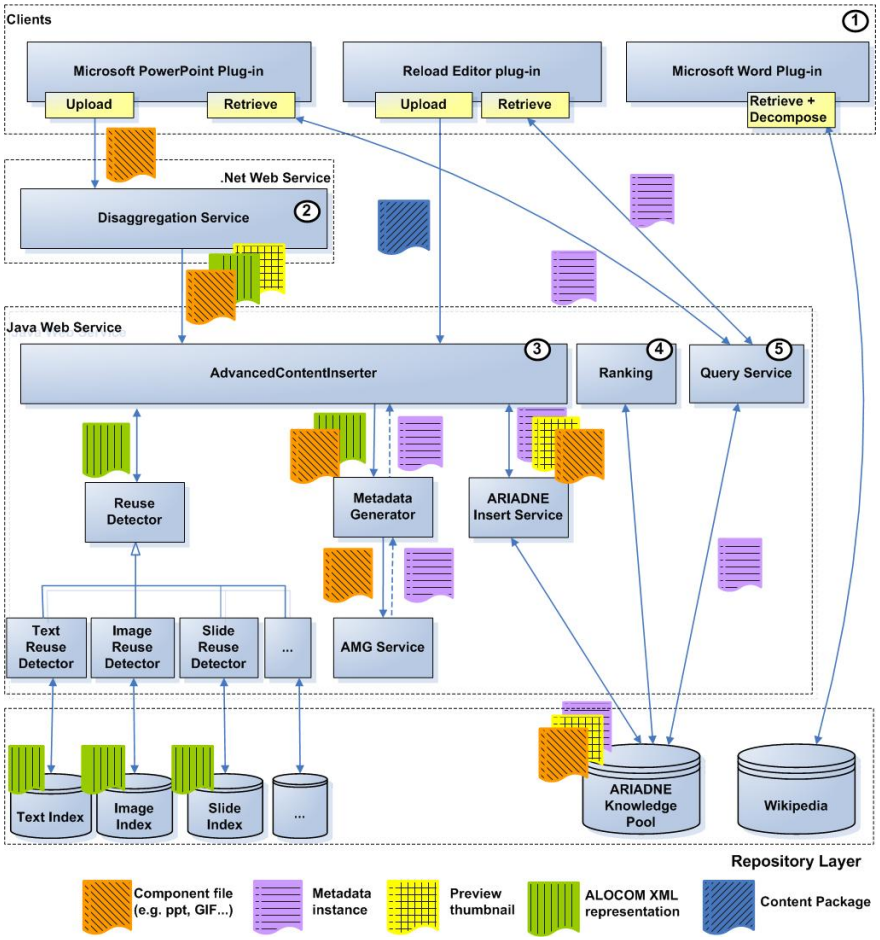

Abbildung 3.5.: Das ALOCOM Framework aus [VOD08]

Im PowerPoint-Plug-in kann eine geöffnete Präsentation auf Wunsch in Komponenten zerlegt und in das ARIADNE-Repository gespeichert werden. Hierbei werden sowohl die einzelnen Folien als Komponenten gespeichert, als auch kleinere Inhalte wie Bilder oder Textblöcke, die auf den Folien enthalten sind. Zudem werden Textinhalte indexiert. Alle Komponenten werden - soweit möglich - mit Metadaten versehen. So erben sie im Fall von PowerPoint beispielsweise den Autor der Präsentation, aus der sie stammen. Beim Speichern in das ARIADNE-Repository werden Duplikate von Komponenten automatisch erkannt und gefiltert. Auf diese Art und Weise wird zudem erkannt, wenn Inhalte wiederverwendet wurden. Die Dekompositionsroutine legt weiterhin explizite Relationen zwischen den einzelnen gespeicherten Komponenten einer Präsentation an. Diese Relationen umfassen einerseits "istTeilVon" (isPartOf) Relationen zwischen einer Komponente und ihren "Kindern" - also zum Beispiel zwischen einer Folie und den darauf enthaltenen Objekten - und andererseits Ordnungsrelationen ("istVorgängerVon") beispielsweise zwischen zwei im Ursprungsdokument aufeinanderfolgenden Folien. Der Definition in dieser Arbeit folgend, handelt es sich bei diesen Relationen demnach um Strukturrelationen (siehe Kapitel 2.4.3). Des Weiteren bietet das PowerPoint-Plug-in eine Suchmaske an, mit der das ARIADNE-Repository nach passenden Komponenten durchsucht werden kann. Hierzu kann nach Schlagworten, aber auch nach Komponententypen gesucht werden. Die gefundenen Komponenten können direkt in die geöffnete

Präsentation integriert werden. Es ist auch möglich, Inhalte, die mit dem Word-Plug-in erzeugt und in ARIADNE hochgeladen wurden, mit dem PowerPoint-Plug-in wiederzuverwenden und umgekehrt.

Das Word-Plug-in ermöglicht die Wiederverwendung von Wikipedia-Inhalten. Hierzu nutzt das Plug-in die in Wikipedia integrierte Suchmaschine, um die gewünschte Wikipedia-Seite aufzufinden. Diese wird dann im Word-Plug-in in Komponenten zerlegt. Die Komponenten werden schließlich in das ARIADNE-Repository hochgeladen. Für die Dekomposition wird die ohnehin vorhandene einheitliche Struktur der Wikipedia-Seiten genutzt. Die Wiederverwendung erfolgt ähnlich wie beim PowerPoint-Plug-in. Es können Schlagworte und der gewünschte Dokumententyp angegeben werden, um Komponenten im Repository aufzufinden. Diese können schließlich in das aktive Dokument eingefügt werden. Eine Dekomposition des Word-Dokuments selbst wird nicht unterstützt. Abbildung 3.5 zeigt das gesamte ALOCOM-Framework.

Auf Basis der Anzahl der Wiederverwendungen einer Komponente und der zugehörigen Metadaten erfolgt ein Ranking der Suchergebnisse beim Retrieval von Komponenten in den Plug-ins. In [KVD06] wurden die im ALOCOM-Repository vorhandenen Komponenten auf ihre Wiederverwendung hin untersucht. Zu diesem Zeitpunkt befanden sich in dem Repository ausschließlich aus PowerPoint-Präsentationen gewonnene Komponenten. Eine Komponente wurde dann als wiederverwendet angesehen, wenn sie bei der Dekomposition eines Foliensatzes schon im Repository vorhanden war. Dies wird mit Hilfe bestimmter Metriken und Fingerprint-Techniken (siehe Kapitel 6) erkannt. Es ergab sich eine Wiederverwendungsverhältnis von 0,22. Was bedeutet, dass im Schnitt 22 Prozent der in einer Präsentation vorhandenen Komponenten wiederverwendet wurden. Die Wiederverwendungsbeziehungen wurden auf unterschiedliche Weise visualisiert, um das Repository den Nutzern zugänglicher zu machen.

Beim ALOCOM-Framework liegt das Hauptaugenmerk auf der Unterstützung der Wiederverwendung mittels Dekomposition und Beschreibung mit objektabhängigen Metadaten, nicht auf der Erfassung von Verwendungs- oder Wiederverwendungsbeziehungen. Nichtsdestotrotz werden in ALOCOM Relationen erfasst. In der Hauptsache handelt es sich dabei um Strukturrelationen, also Relationen, die die ursprüngliche Struktur eines, in Komponenten zerteilten, Dokuments widerspiegeln. Wiederverwendungsbeziehungen werden mit Hilfe von Post-Processing-Algorithmen auf Komponentenebene erfasst. Dabei gehen im Vergleich zu einem Ansatz, der die Beziehungen während der Entstehung erfasst, Informationen verloren. So kann beispielsweise nicht mit Sicherheit gesagt werden, welches Dokument die Quelle und welches das Ziel einer Relation ist. Es können nur Aussagen über einzelne Komponeten getroffen werden. Wenn eine Komponente in 30 Dokumenten wiederverwendet wird, sagt diese Information nichts über die Beziehungen der Dokumente untereinander aus, welche die Komponenten wiederverwenden. Des Weiteren wird eine Wiederverwendung nur erkannt, wenn die entsprechenden Komponenten auch in das ALOCOM-Repository eingestellt werden.

TeNDaX

TeNDaX [HBD04] ist die Abkürzung für Text Native Database eXtension. Es handelt sich dabei um ein datenbankbasiertes Textverarbeitungswerkzeug zur kollaborativen Erstellung von Dokumenten. Jede Aktion im TeNDaX-Editor, dazu zählen beispielsweise das Hinzufügen und Löschen von Worten, Layout-Änderungen, Copy&Paste-Aktionen oder das Einfügen von Tabellen oder Bildern, wird in Transaktions-

form in einer Datenbank gespeichert. Somit können alle Aktionen aller Autoren des Dokuments zurückverfolgt und entsprechend behandelt werden. Zusätzlich zu den Transaktionen werden zu jeder Aktion eines Nutzers Metadaten über diese Aktion erfasst. Dies geschieht auf unterschiedlichen Ebenen eines Dokuments: auf Dokumentenebene, Abschnittsebene und Buchstabenebene.

Auf Buchstabenebene werden unter anderem der bearbeitende Autor, Datum und Zeit, Informationen zum Redo- und Undo-Status oder auch Copy&Paste-Referenzen als Metadaten gespeichert. Die erfassten Metadaten werden in TeNDaX größtenteils dazu benutzt, die komplexen Prozesse, die beim Bearbeiten eines Dokuments durch verschiedene Autoren entstehen, zu verwalten. Hierdurch kann beispielsweise eine globale Undo- und Redo-Funktion implementiert werden.

In [HHD05] wird beschrieben, wie diese Metadaten zusätzlich genutzt werden können, um ein dynamisches Dokumentenmanagementsystem als Alternative zu hierarchischen, statischen Dateisystemen umzusetzen. Teil des Dokumentenmanagementsystems ist auch eine Applikation, welche die durch Copy&Paste entstandenen Relationen zwischen Dokumenten visualisiert.

In TeNDaX wird das umgesetzt, was in den meisten breit genutzten Autorenwerkzeugen fehlt: Die Informationen, die während der Bearbeitung von Dokumenten anfallen, werden konsequent erfasst und zur Unterstützung der Nutzer und Erweiterung der Funktionalitäten des Werkzeugs eingesetzt. Zu den Informationen, die erfasst werden, zählen auch Wiederverwendungsrelationen, die durch Copy&Paste-Aktionen entstanden sind, was dem Typ 2 der Kategorisierung nach Libbrecht entspricht (siehe Abschnitt 2.2.2). Weitere Relationen werden jedoch nicht erfasst. Außerdem ist die Erfassung an das Werkzeug selbst gebunden. Somit verbleiben die gesammelten Informationen im System. Dadurch, dass die Erfassung der Informationen aus dem Quellcode heraus erfolgt, ist eine Übertragbarkeit das Ansatzes auf andere Werkzeuge, zum Beispiel solche mit weiterer Verbreitung wie Microsoft PowerPoint oder Microsoft Word nicht gegeben.

docendo

docendo (ehemals RecourceCenter) [HHRS05] ist eine Kombination aus Repository und webbasiertem Autorenwerkzeug zur Unterstützung des Authoring-by-Aggregation von Lernressourcen. Das Repository ermöglicht ein Aufladen modularer Inhalte, wie Bilder, Animationen, Videos oder Audiodateien. Diese werden halbautomatisch mit Metadaten versehen und können so auffindbar gemacht werden. Weiterhin bietet docendo Nutzern die Möglichkeit, E-Learning-Kurse zu erstellen. Dabei bilden sie zunächst eine logische Struktur, indem sie einen Kurs in Abschnitte und Unterabschnitte gliedern. Diese können dann mit Hilfe eines integrierten Editors mit Inhalten gefüllt werden. Die Nutzer können jederzeit das Repository nach Inhalten - den aufgeladenen modularen Inhalten, aber auch nach bereits vorhandenen Kursen oder Abschnitten - durchsuchen und diese wiederverwenden. Hierdurch entstehen eine große Zahl von Wiederverwendungsbeziehungen, die wiederum zur Unterstützung der Auffindbarkeit der Inhalte im Repository genutzt werden können. In [LRS07] wird beschrieben, welche Informationen erfasst werden und wie diese genutzt werden können. Abbildung 3.6 zeigt die Detailansicht eines Abschnitts, einschließlich der erfassten Relationen.

Die Erfassung und Nutzung von Lebenszyklusinformationen in docendo diente als Testmuster und Vorstufe dieser Arbeit. Als Kombination aus Autorenwerkzeug und Repository vereint es sowohl die

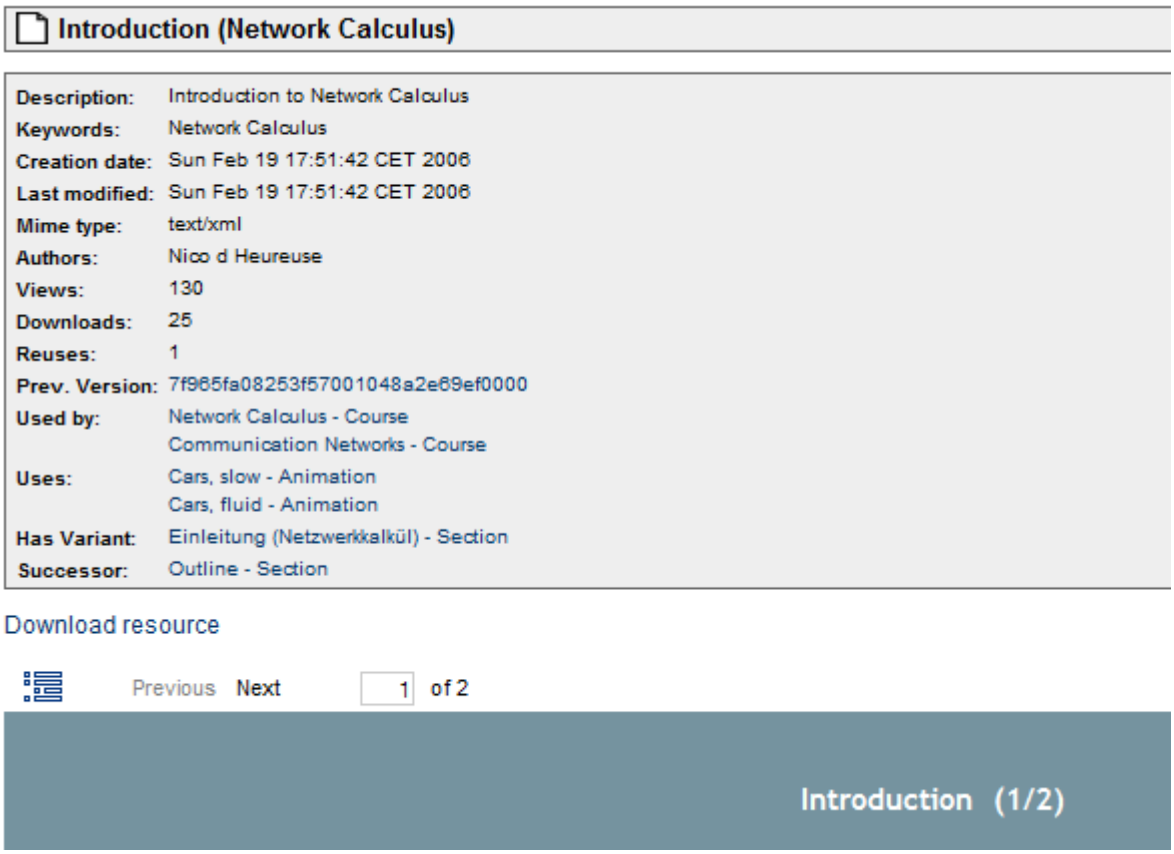

Abbildung 3.6.: Kursabschnitt mit Beziehungsinformationen in docendo

Erstellungs- und Überarbeitungsphase als auch die Zugriffs- bzw. Bereitstellungsphase des Lebenszyklus. Somit können die gewonnenen Informationen ohne großen Aufwand direkt genutzt werden, beispielsweise zum Verbessern der Suchfunktion des Repositories. Durch die Ausrichtung des docendo-Systems auf Lernressourcen, und hierbei Authoring-by-Aggregation als speziellen Autorenprozess, ist eine Übertragbarkeit des Konzepts auf andere Systeme nicht gegeben. Des Weiteren verbleiben die gesammelten Informationen in docendo und werden nicht über die Systemgrenzen hinweg transportiert, um außerhalb genutzt werden zu können. Mit Hilfe dieses Ansatzes konnte jedoch gezeigt werden, dass eine Erfassung von Lebenszyklusinformationen prinzipiell möglich und sinnvoll ist [LRS07].

Beziehungsinformationen für die Desktop-Suche und Persönliches Informationsmanagement

Desktop-Suchmaschinen sind Suchmaschinen, die auf die Suche in Dateisystemen auf lokalen Rechnern spezialisiert sind. In einigen Ansätzen zur Desktop-Suche werden Beziehungsinformationen erfasst und genutzt, um die Funktionalität der Suchmaschine zu verbessern. Chirita et al. machen sich die Verwendung von Dokumenten und deren Kontext zunutze, um daraus Beziehungsinformationen abzuleiten, die für die Desktop-Suche nutzbar sind. In [CGG+05] werden drei Quellen für solche Informationen genannt, und es wird beschrieben, wie sie genutzt werden können:

- Der **E-Mail-Kontext**: Dokumente, die als E-Mail Anhang gesendet werden, verlieren ihren inhaltlichen Zusammenhang mit der zugehörigen E-Mail, wenn sie abgespeichert werden. Chirita et al.

schlagen vor, diesen Zusammenhang zu erhalten, indem beim Abspeichern des Dokuments eine Beziehung zwischen der Mail und dem Dokument hergestellt wird. Hierfür wird ein RDF-Schema vorgestellt, mit dessen Hilfe ein Dokument wieder seinem ursprünglichen Kontext zugeordnet werden kann.

- **Ordner-Hierarchien**: Viele Nutzer investieren Zeit und Mühe in den Aufbau von Ordner-Hierarchien, in denen sie ihre Dokumente ablegen. Dies dient ihnen dazu, die Auffindbarkeit der Dokumente zu verbessern. Oftmals werden dabei implizit Taxonomien erstellt. Chirita et al. machen diese Informationen für eine semantische Suche nutzbar, indem die Dokumente innerhalb der Ordner der entsprechenden Kategorie zugeordnet werden. WordNet [Fel98] wird dabei benutzt, um Synonyme, Hypernyme und Holonyme zu identifizieren und entsprechend dem entworfenen RDF-Schema auszuzeichnen.

- **Browser-Cache**: Hierbei wird der Cache des Browers genutzt, um Seiten anzuzeigen, die zu einer gefundenen Seite in Verbindung stehen. Das können zum Beispiel kürzlich aufgerufene Unterseiten der angezeigten Seite sein. Konzeptionell sollen jedoch auch implizite Relationen, wie zum Beispiel wissenschaftliche Referenzen, einbezogen werden.

Beim ersten der drei Fälle wird eine Lebenszyklusinformation während der Entstehung - dem Speichern des Anhangs einer Mail - erfasst. In den beiden anderen Fällen sind die gewonnenen Informationen rekonstruierbar, da die Prozesse, die zur Entstehung der Informationen geführt haben, zum Zeitpunkt der Auswertung bereits abgeschlossen sind.

In [CN06] werden die in [CGG+05] vorgestellten Konzepte um ein weiteres, allgemeiner verwendbares erweitert. Hier werden Beziehungen zwischen Dokumenten aus Zugriffen, die innerhalb eines kurzen Zeitraums auf diese erfolgen, geschlossen. Es werden also Beziehungsinformationen aus Verwendungsinformationen abgeleitet, demnach indirekte Beziehungsinformationen erfasst. Immer wenn zwei Dokumente in kurzer Folge geöffnet werden, wird aus dieser Zugriffssequenz eine semantische Verwandtschaft zwischen diesen Dokumenten geschlossen [Chi07]. Diese Technik wird auch von Okoli et al. in [OB09] genutzt, um Relationen zwischen im Dateisystem zugegriffenen Dokumenten abzuleiten.

In *Ivan* [PM08], einem System zur Unterstützung des persönlichen Informationsmanagements (PIM), werden ebenfalls Beziehungen zwischen Dokumenten erfasst, wenn auf diese innerhalb kurzer Zeiträume zugegriffen wird. Die erfassten Beziehungen werden zudem stärker gewichtet, wenn weitere Aktionen zwischen den beteiligten Dokumenten erfolgen. Dazu zählen auch Copy&Paste-Aktionen zwischen Dokumenten.

In [BCN06] wird *jNotify*, ein System zur applikationsunabhängigen Erfassung solcher Metadaten, vorgestellt. Eine im Hintergrund laufende Applikation beobachtet das Nutzerverhalten und kann auf Ebene des Betriebssystems, unabhängig von der verwendeten Applikation, aus der die Daten stammen, sowie unabhängig von der Applikation, in der die Daten genutzt werden, Informationen erfassen. jNotify funktioniert auf Basis von Ereignissen, die vom Betriebssystem auf Dateiebene ausgelöst werden, und findet in drei Kontexten Anwendung: Zur Erfassung des Kontextes von E-Mails, Gewinnung von Informationen aus Ordner-Hierarchien und zur Erfassung des Kontextes von Publikationen.

Auch auf dem Gebiet der Semantic-Desktops [SBD05] werden Informationen aus der Aktivität des Nutzers erfasst, um so die Funktionalität von Semantic-Desktop-Anwendungen wie Gnowsis [Sau04],

Nepomuk [GHM+07] oder Haystack [KBH+05] zu unterstützen. Dabei werden, ähnlich wie bei der beschriebenen Unterstützung der Desktop-Suche, Informationen und Beziehungen aus geloggten Nutzeraktivitäten gewonnen.

hylOs

Das E-Learning Content Management System hylOs (hypermedia learning Object system) [EKR+03] unterstützt einen konstruktivistischen Lernansatz. Die Lerninhalte werden nicht linear dargeboten, vielmehr kann der Lernende unterschiedlichen, ihm dargebotenen Links folgen. Diese werden kontextsensitiv offeriert. Um dies bewerkstelligen zu können, sind semantische Relationen zwischen verschiedenen Lernobjekten innerhalb der Plattform notwendig. Diese werden halbautomatisch bei der Einstellung neuer Inhalte erstellt. Dazu gibt es in hylOs zwei Möglichkeiten: Zum einen können bestimmte Relationen aus den Metadaten oder der Struktur abgeleitet werden ("istTeilVon", "istVerwandtMit",...), zum anderen kann die Menge vorhandener Beziehungen mit Hilfe eines algebraischen Regelwerks erweitert werden. Dies geschieht mit Hilfe einer "Schlußfolgerungs-Engine", die die neuen Relationen herleitet [EHS05]. So kann als einfaches Beispiel aus den bestehenden Relationen "A ist Teil von B" und "B ist Teil von C" auch "A ist Teil von C" gefolgert werden. Die Relationen werden in hylOs jedoch nicht kontinuierlich aus dem Lebenszyklus der Lernobjekte heraus erfasst, sondern beim Einstellen der Inhalte erzeugt.

3.1.3 Zusammenfassung

In den vorangegangenen Unterkapiteln wurden die wichtigsten verwandten Ansätze zur Erfassung von Verwendungs- und Beziehungsinformationen betrachtet. Bei den Ansätzen zur Erfassung von Verwendungsinformationen fällt auf, dass die meisten Ansätze die Informationen nicht dokumenten-, sondern nutzerzentriert erfassen. Dadurch wird eine komplette Abdeckung des Lebenszyklus eines Dokuments und seiner Instanzen erschwert. Der *Ecological Approach* von Brooks und McCalla konzentriert sich völlig auf die Nutzungsphase. Die Ansätze aus APOSDLE und DYONIPOS erfassen lediglich den Kontext eines Nutzers; dies überschneidet sich teilweise mit Lebenszyklusinformationen der Dokumente, die der Nutzer verwendet, deckt deren Lebenszyklus jedoch keinesfalls ab. Die Ansätze aus dem Bereich der kontextsensitiven Dienste dienen im Allgemeinen nicht der Erfassung von Informationen über Dokumente, wurden jedoch der Vollständigkeit halber aufgenommen.

Beziehungsinformationen werden nur vereinzelt während ihrer Entstehung erfasst. In den beiden Systemen, die direkte Beziehungsinformationen während der Entstehung erfassen - docendo und TeNDaX - verbleiben die Informationen im System. Es gibt jedoch einige Ansätze, welche die Beziehungsinformationen mit Benutzerunterstützung erfassen (hylOs) oder sie im Nachhinein gewinnen (ALOCOM). Es gibt einige Ansätze im Bereich des Personal Information Managements (PIM), die sich mit der Erfassung indirekter Relationen, also aus der Folgerung von Beziehungen aus Verwendungsinformationen, befassen. Verschiedene weitere Ansätze befassen sich am Rande mit Beziehungsinformationen. In [Mue06] wird ein Ansatz zur konsistenten Änderungsverwaltung von strukturierten wie unstrukturierten Dokumenten vorgestellt. Es handelt sich dabei um ein erweitertes Versionierungssystem. Hierzu wurde ein spezielles XML-Modell entwickelt mit dessen Hilfe auch Relationen innerhalb und zwischen Dokumen-

ten erfasst werden können. Dies geschieht jedoch auf explizite Aktionen der Nutzer hin (Check-In / Check-Out). Der Ansatz zielt nicht darauf ab, Informationen aus dem Lebenszyklus der Dokumente kontinuierlich zu sammeln. Es gibt darüber hinaus diverse Ansätze, Beziehungen zwischen Dokumenten mit Hilfe von Post-Processing-Techniken zu erkennen. Häufig werden dabei Algorithmen verwendet, die auch zur Erkennung von Plagiaten eingesetzt werden. Ein wichtiger Anwendungsfall solcher Techniken ist die Rückverfolgung von Wiederverwendung einzelner Teile von Dokumenten im Internet sowie die Filterung von Nahezu-Duplikaten durch Suchmaschinen wie Google oder Yahoo. Diese Techniken werden in den Kapiteln 5 und 6 ausführlich behandelt und diskutiert.

3.2 Ansätze und Systeme zur Nutzung von Lebenszyklusinformationen

Wie auch bei den Ansätzen zur Erfassung von Lebenszyklusinformationen, sind die Ansätze zur Nutzung von Lebenszyklusinformationen hauptsächlich auf Verwendungsinformationen und weniger auf Beziehungsinformationen fokussiert. Es existieren jedoch auch einige Ansätze, die Beziehungsinformationen zwischen Nutzern oder Dokumenten berücksichtigen. Die Ansätze zur Nutzung von Verwendungs- und Beziehungsinformationen lassen sich in zwei größere Teilbereiche unterteilen, die im Folgenden beschrieben werden: (1.) Ansätze aus dem Bereich des E-Learning und Wissensmanagements sowie (2.) aus dem Bereich des Information Retrieval.

E-Learning & Wissensmanagement

Der Ursprung der Erfassung von Verwendungsinformationen - *der Ecological Approach* - verfolgt das Ziel, durch das Sammeln von Informationen über einen Lernenden diesen zu modellieren, um ihm so angemessene Lernressourcen, Lerngruppen oder passende Lernpfade durch verschiedene Lernobjekte vorschlagen zu können und ihn mit anderen Lernenden zu vernetzen [McC04].

Die erfassten Informationen können jedoch auch dazu verwendet werden, um beispielsweise bestehende Metadaten zu verbessern. So werden in [WMND06] Verwendungsinformationen aus einem Learning Management System (LMS) rückgekoppelt, um Metadaten der zugehörigen Lernobjekte in einem Repository zu aktualisieren. Dadurch können zum Beispiel Felder wie *Dauer* oder *Schwierigkeit* mit den realistischen, im LMS gesammelten Informationen abgeglichen werden.

Auch in Unternehmen ist ein Einsatz von Verwendungsinformationen interessant. Wenn Attention Metadata - also Verwendungsinformationen - in Human Ressource Management, Wissensmanagement oder Workflow-Systemen erfasst werden, können sie dafür genutzt werden, um die Lernprozesse der Mitarbeiter zu steuern und die Ziele von Unternehmen und Mitarbeitern diesbezüglich in Einklang zu bringen [WMND06]. Auch für die Entscheidungsfindung in Unternehmen können Verwendungsinformationen hilfreich sein. Die Verwendungsinformationen werden erfasst und gehen als zusätzliche Metadaten in die Systeme zur Unterstützung von Entscheidungsfindungsprozessen ein [AD06].

Ein weiterer Anwendungsfall von Verwendungsinformationen liegt auf der Hand: In [TSQ07] werden diese Informationen dafür genutzt, Statistiken über die Nutzung, Wiederverwendung und Teilnahme an Online-Konferenzen zu gewinnen und auszuwerten.

In den meisten gängigen e-Commerce-Plattformen gibt es Systeme, die einem Käufer weitere interessante Artikel vorschlagen (siehe [SKR01], [GG03]). Ein sehr bekanntes Beispiel hierfür ist die Verkaufsplattform "Amazon". Die Vorschläge basieren auf Daten über den Nutzer. Bei diesen Daten handelt es sich prinzipiell um Verwendunginformationen, die durch Interaktion eines Nutzers bzw. Kunden mit dem System entstehen. Oftmals kommen in solchen Plattformen *kollaboratives Filtern* [GSK+99], [BHK98] oder *Cluster-Modelle* [UF98] zum Einsatz. In beiden Fällen werden die über einen Nutzer gesammelten Informationen mit denen anderer Nutzer verglichen, um Nutzer mit ähnlichen Interessen zu finden. Wenn ein solcher Nutzer gefunden wurde, werden dem Käufer die Produkte aus dem Profil des gefundenen Nutzers vorgeschlagen. Amazon selbst verwendet jedoch einen leicht abgewandelten Ansatz. Hier werden Vorschläge nicht auf Basis der Profile anderer Käufer durch kollaboratives Filtern, sondern allein auf Basis der Produkte selbst generiert [LSY03]. Dies geschieht, indem Listen erstellt werden mit Produkten, die zueinander passen. Ein Produkt wird als zu einem anderen passend klassifiziert, wenn ein Käufer die beiden Produkte gemeinsam erstanden hat.

Ein weiteres großes Anwendungsgebiet für sowohl Beziehungs- als auch Verwendungsinformationen ist die Suche und das damit verbundene Ranking der Suchergebnisse. In [CGG+05] wird beschrieben, wie die Informationen, die aus der Ordner-Hierarchie im Dateisystem, dem Zusammenhang von E-Mails mit ihren Anhängen und dem Browser-Cache gewonnen werden, für die Suche nutzbar gemacht werden können. Dies geschieht vor allem durch Anreichern von Suchergebnissen, indem zusätzliche Felder, die durch die oben genannten Mechanismen gewonnen wurden, ebenfalls durchsucht werden.

In Daffodil [KFKS05], einem Zugriffssystem für heterogene Digitale Bibliotheken, werden Informationen über Nutzer-Aktionen und somit Verwendungsinformationen dazu genutzt, Relevance-Feedback zu erzeugen. Anhand der Klicks eines Nutzers nach einer Suchanfrage an das System wird ermittelt, welche der angezeigten Ergebnisse für den Nutzer eine hohe Relevanz haben und welche weniger [EKH09]. Dementsprechend kann das System zukünftige Suchergebnise entsprechend anpassen. Diese Verwendungsinformationen sind auch Teil des Logging-Konzepts von Daffodil, das die Erfassung von Verwendungsinformationen auf unterschiedlichen Ebenen, vom Tastaturanschlag bis hin zum Suchverhalten eines Nutzers vorsieht [KH09]. Neben der oben genannten Art der Nutzung können die Informationen unter anderem dazu genutzt werden, verschiedene Digitale Bibliotheken gegeneinander zu evaluieren [KFK+06], die Benutzeroberfläche dem Erfahrungsstand eines Nutzers anzupassen, an den Nutzer angepasste Suchstrategien vorzuschlagen oder das Retrieval selbst durch Re-Ranking oder Erweiterung der Suchergebnisliste zu verbessern [KKH08].

Die impliziten Beziehungen zwischen Dokumenten, die sich aus der Überwachung von Zugriffssequenzen auf Desktop-Dokumenten ergeben, werden dazu genutzt, um Suchergebnisse zu ranken. In [CN06] zeigen Chirita und Nejdl, dass eine Kombination aus der TF-IDF-Methode [Jon72] und dem auf den gewonnenen Beziehungen basierenden Ranking-Algorithmus gute Ergebnisse erzielt. In [GCC+08] werden weitere Evaluationen des zugriffssequenzbasierten Ranking-Ansatzes vorgestellt. Sowohl die Ergebnisse des Rankings über die Zugriffssequenz, als auch die beschriebene Anreicherung von Suchergebnissen durch Verwendungsinformationen und Beziehungen werden in [Chi07] ausführlich erläutert. Beagle++

[BCC+06] ist eine Suchmaschine für die Desktop-Suche, die als Plattform für die Umsetzung der durch Chirita et al. entwickelten Konzepte und Funktionen dient [CCNP06].

Einen weiteren Ansatz zum Ranking stellen Ochoa und Duval in [OD06] vor. Auf der Basis von Daten aus Contextualized-Attention-Metadata (CAM) wird ein Graph konstruiert, mit dessen Hilfe Lernressourcen gerankt werden können. Auch hier werden Beziehungen zwischen Benutzern, Autoren, Kursen und Lernobjekten anhand der in CAM gespeicherten Zugriffsinformationen hergeleitet. Aus den sich ergebenden Graphen können dann Popularitätsfaktoren hergeleitet und die Objekte entsprechend sortiert werden.

3.3 Zusammenfassung und Diskussion der Ansätze

Kapitel 3.2 hat gezeigt, dass es vielfältige Möglichkeiten gibt, Lebenszyklusinformationen zu nutzen und so einen Mehrwert für Autoren oder Nutzer zu erzeugen. Der Fokus dieser Arbeit liegt jedoch nicht auf der Nutzung der Informationen, sondern auf deren Erfassung und Verwaltung. Deshalb werden im Folgenden die vorhandenen Ansätze zur Erfassung von Lebenszyklusinformationen betrachtet, verglichen und anhand der in Kapitel 2.5 gesammelten Kriterien beurteilt. Abbildung 3.7 zeigt die vorgestellten Ansätze im Vergleich.

- nicht vorhanden x vorhanden (x) eingeschränkt vorhanden	Ecological Approach	CAM-Framework	APOSDLE / DYONIPOS	Kontextsensitive Dienste	ALOCOM-Framework	TeNDaX	docendo	SHAMAN	Desktopsuche & PIM	hylOs	LIS.KOM
Erfassung von Verwendungsinformationen	x	x	x	-	-	x	(x)	x	x	-	(x)
Erfassung von Beziehungsinformationen	-	-	-	-	x	x	x	(x)	(x)	x	x
Erfassung während der Entstehung	x	x	x	x	-	x	x	(x)	(x)	-	x
Abdeckung des gesamten Lebenszyklus	-	(x)	-	-	-	-	-	-	-	-	x
Unterstützung allgemeiner Autorenprozesse	-	-	-	-	x	-	-	-	-	-	x
Überbrückung von Systemgrenzen	x	x	-	x	(x)	-	-	(x)	(x)	-	x
Übertragbar auf andere Applikationen / Systeme	x	x	-	x	x	-	-	(x)	(x)	-	x

Abbildung 3.7.: Vergleich der Verwandten Ansätze

Auf den ersten Blick fällt auf, dass es keinen Ansatz gibt, der sowohl Verwendungs- als auch Beziehungsinformationen abdeckt. Die Ansätze aus dem Bereich der kontextsensitiven Dienste und APOSDLE und DYONIPOS sind zwar verwandt, verfolgen jedoch insgesamt andere Ziele. Ähnliches gilt für das ALOCOM-Framework. Hier werden zwar Beziehungsinformationen erfasst, jedoch aus der Struktur der in das System aufgeladenen Dokumente und nicht aus deren Lebenszyklus. Der Ecological Approach

beschränkt sich allein auf die Erfassung von Verwendungsinformationen während der Nutzungsphase. In SHAMAN ist die Erfassung von Lebenszyklusinformationen unterschiedlicher Art und in verschiedenen Szenarien zwar Teil des Konzepts, wurde bisher jedoch weder konkretisiert noch operationalisiert. Zudem ist hier die Erfassung von direkten Beziehungsinformationen nicht vorgesehen. Das CAM-Framework kommt der in dieser Arbeit vorgestellten Idee am nächsten. Es bietet ein systemübergreifendes Framework zur Erfassung von Verwendungsinformationen und ist konzeptuell auf beliebige Applikationen übertragbar. Auch stellt es ein offenes Format zur Verwaltung und Übertragung der gesammelten Informationen zur Verfügung. Beziehungsinformationen sind im CAM-System jedoch nicht vorgesehen. Auch bei der Abdeckung des gesamten (logischen) Lebenszyklus eines Dokuments gibt es Einschränkungen: Die Informationen werden im CAM-Framework nutzerzentriert und nicht dokumentenzentriert gesammelt. Es kann zwar eine dokumentenzentrierte Sicht auf die gesammelten Informationen berechnet werden, dies deckt aber nicht notwendigerweise auch den ganzen Lebenszyklus eines Dokuments ab. Der Weg eines Dokuments durch die Hände verschiedener Nutzer und über verschiedene Systeme wird vom CAM-Framework nicht abgedeckt, da die Verbindungen, also Beziehungen, zwischen den physikalischen Instanzen eines Dokuments nicht erfasst werden und das CAM-System jede Instanz eines Dokuments als unabhängiges Dokument betrachtet. Vielmehr werden Informationen über Applikationen, Dokumente und Systeme, die ein Nutzer verwendet, gesammelt.

Es gibt allgemein kaum Ansätze, die bereits Beziehungsinformationen aus dem Lebenszyklus von Dokumenten erfassen. Bei beiden Systemen, in denen direkte Relationen während der Entstehung erfasst werden - docendo und TeNDaX - verbleiben die Informationen im jeweiligen System. Auch eine Übertragbarkeit des Ansatzes auf andere Applikationen ist nicht gegeben. Zudem handelt es sich in beiden Fällen um Applikationen, die sehr spezifische Anwendungsszenarien unterstützen. Des Weiteren ist der Lebenszyklus in beiden Fällen nicht abgedeckt, auch die erfassten Beziehungsinformationen umfassen nicht alle in Kapitel 2.4.3 vorgestellten Relationstypen. Die vorgestellten Ansätze aus dem Bereich der Desktopsuche und des persönlichen Informationsmanagements sehen eine Erfassung von Beziehungsinformationen aus dem Lebenszyklus zumindest teilweise vor. Zumeist handelt es sich bei den Beziehungsinformationen um indirekte Beziehungsinformationen, die zum Beispiel aus Zugriffen im Dateisystem abgeleitet wurden. Auch hier ist eine Abdeckung des Lebenszyklus eines Dokuments nicht gegeben, und die erfassten Informationen verbleiben in den meisten Fällen auf dem System eines Nutzers und werden nicht systemübergreifend gesammelt und verwaltet. In hylOs werden die Relationen in Interaktion mit dem Nutzer erzeugt und danach auf Basis eines Regelwerks angereichert. Eine Erfassung von Lebenszyklusinformationen findet hier also nicht statt.

Durch das CAM-System ist die Erfassung und Verwaltung von Lebenszyklusinformationen, was Verwendungsinformationen betrifft, schon ausreichend abgedeckt - auch wenn der Ansatz noch Schwächen in Bezug auf die Abdeckung des Lebenszyklus von Dokumenten aufweist. Deshalb konzentriert sich das im folgenden Kapitel vorgestellte LIS.KOM-Framework auf die Erfassung von Beziehungsinformationen aus dem Lebenszyklus von Wissensdokumenten.

4 Erfassung, Verwaltung und Nutzung von Lebenszyklusinformationen

Die Analyse der verwandten Arbeiten hat gezeigt, dass insbesondere im Bereich der Erfassung und Nutzung von Beziehungsinformationen noch keine systemübergreifenden und generischen Ansätze vorhanden sind. Daher liegt das Hauptaugenmerk in dieser Arbeit auf Beziehungsinformationen. Verwendungsinformationen werden jedoch nicht grundsätzlich ausgeschlossen, sondern vielmehr im Hinblick auf ihre Verwendbarkeit in Kombination mit Beziehungsinformationen betrachtet. Dieses Kapitel widmet sich dem modularen Konzept, das diese Arbeit verfolgt. Dieses unterteilt sich in drei Hauptschritte. Zunächst werden Lebenszyklusinformationen gewonnen, daraufhin müssen sie vorgehalten und verfügbar gemacht werden. Schließlich können die Informationen genutzt werden. Demzufolge ergeben sich die folgenden drei Unterkonzepte *Erfassung*, *Verwaltung* und *Nutzung*, die mit den drei Hauptabschnitten dieses Kapitels korrespondieren. Abbildung 4.1 zeigt die Bestandteile des Gesamtkonzeptes. Die theoretische Basis bilden das in Kapitel 2 beschriebene Lebenszyklusmodell und die darauf aufbauende Analyse und Kategorisierung von Lebenszyklusinformationen. Neben den genannten Aufgaben der Erfassung, Verwaltung und Nutzung der Informationen enthält das Konzept auch noch eine Komponente, die die Validierung von Lebenszyklusinformationen vorsieht. Die hierfür entwickelten Algorithmen werden in den Kapiteln 5 und 6 erläutert. Auch hier liegt das Hauptaugenmerk auf der Erfassung sowie der Verwaltung von Lebenszyjlusinformationen. Es werden im weiteren Verlauf des Kapitels zwar unterschiedliche Nutzungsszenarien vorgestellt, diese dienen jedoch nur als Beispiel und Hinweis auf das Nutzungspotential der erfassten Informationen.

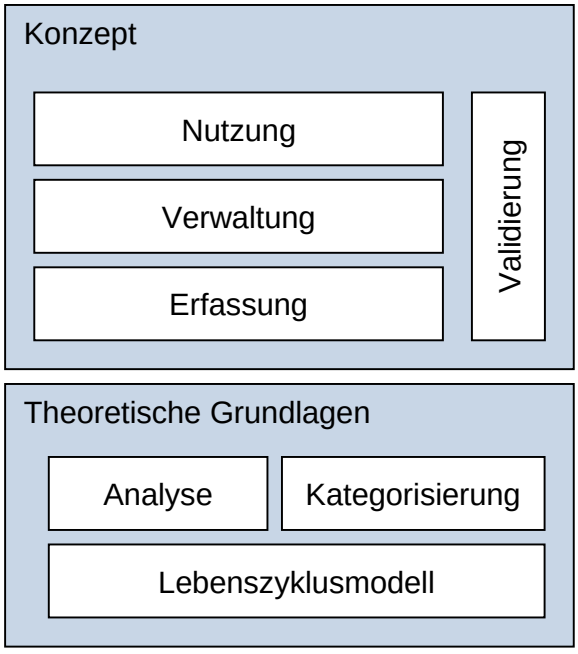

Abbildung 4.1.: Komponenten des Konzeptes

4.1 Erfassung

Wie in Kapitel 2.4 beschrieben sind Lebenszyklusinformationen prozessorientiert. Um Lebenszyklusinformationen zu erfassen, ist es folglich notwendig, die Prozesse, aus denen heraus die Informationen entstehen, zu überwachen. Die Prozesse wiederum, die zur Entstehung von Lebenszyklusinformationen führen, sind an bestimmte Applikationen gebunden und werden dort ausgeführt. Da diese Applikationen im Allgemeinen jedoch keine Lebenszyklusinformationen erfassen, ist es notwendig, die Applikationen zu erweitern, um eine Erfassung der Informationen zu ermöglichen. Dies kann beispielsweise in Form von Plug-ins für die Applikationen geschehen. Die folgenden Abschnitte identifizieren die Herausforderungen, die bei der Erfassung von Lebenszyklusinformationen gemeistert werden müssen und zeigen Lösungswege, sowohl für die Erfassung von Beziehungsinformationen als auch die Erfassung von Verwendungsinformationen auf.

4.1.1 Erfassung und Überwachung von Beziehungsinformationen

Die nachfolgenden Betrachtungen beschränken sich auf Wiederverwendungsbeziehungen; indirekte Beziehungen, wie in Abschnitt 2.4.3 beschrieben, werden hier nicht berücksichtigt.

Analyse der Herausforderungen

Um Beziehungsinformationen aus dem Lebenszyklus von Dokumenten erfassen zu können, müssen verschiedene Voraussetzungen erfüllt sein. Diese werden kurz zusammengefasst, bevor sie eingehend erläutert werden:

1. Ereignisse, die zur Entstehung einer Relation führen, müssen erkannt werden.
2. Alle Informationen, die zur Beschreibung einer vollwertigen Relation notwendig sind, müssen erfasst werden.
3. Aktionen auf Quell- oder Zieldokument, die dazu führen können, dass Relationen ihre Validität verlieren, müssen verfolgt und bewertet werden.
4. Aktionen, die außerhalb der überwachten Applikationen erfolgen, müssen verfolgt und entsprechend berücksichtigt werden.
5. Aktionen, die auf nicht überwachten Systemen erfolgen, müssen erkannt und bewertet werden.

Zu 1): Wiederverwendungsrelationen entstehen meist durch bestimmte Aktionen eines Nutzers innerhalb einer bestimmten Applikation. Welche Aktionen das sind, ist vom Typ der Relation abhängig. Abhängig vom Typ der Wiederverwendungsrelation ergeben sich unterschiedliche Ereignisse und Aktionen, die verfolgt werden müssen:

- *Variantenrelationen*: Variantenrelationen können auf zwei unterschiedliche Arten entstehen: Entweder wird ein bestehendes Dokument mit Hilfe einer Applikation unter einem anderen Namen

gespeichert oder es wird außerhalb dieser Applikation eine Kopie des Dokuments erstellt, zum Beispiel, indem es im Dateisystem in einen anderen Ordner kopiert wird.

- *Aggregationsrelationen & Elementrelationen*: Diese Relationen entstehen, wenn der Nutzer ein Element aus einem Dokument oder das Dokument selbst in ein weiteres Dokument einfügt. Zumeist geschieht dies, indem ersteres kopiert und dann in das Zieldokument eingefügt wird (Copy & Paste). Hierzu können Tastaturkürzel oder die Benutzerschnittstelle der Applikation benutzt werden. Dabei muss sowohl die Aktion des Kopierens als auch die des Einfügens von der Erfassungskomponente verfolgt werden. Oft ist es auch möglich, ein Objekt durch Ziehen mit der Maus in ein anderes Dokument einzufügen (Drag&Drop).

- *Verknüpfungsrelationen:* Verknüpfungsrelationen entstehen meistens durch spezielle Prozesse innerhalb der verwendeten Applikation. Inwiefern diese verfolgbar sind, hängt zumeist von der Schnittstelle der Applikation ab. Oft werden Verknüpfungen mit Hilfe von Hyperlinks bewerkstelligt. In einem solchen Fall muss das Einfügen oder Erstellen von Hyperlinks verfolgt und das Ziel des Links entsprechend erfasst werden.

In Tabelle 4.1 sind die Ereignisse, die zur Entstehung einer Relation führen, zusammengefasst. Über diese speziellen Ereignisse hinaus müssen auch noch eine Reihe von grundlegenden Ereignissen erkannt werden, um die Funktionalität des Ansatzes zu gewährleisten: Hierzu zählen das Öffnen, Speichern und Schließen eines Dokuments sowie sein Aktivitätszustand - also, ob das Dokument gerade den Fokus hat oder nicht. Die beschriebenen Ereignisse treten fast ausschließlich innerhalb der Applikation auf, die zum Überarbeiten des Dokuments genutzt wird und somit in der Erstellungs- und Überarbeitungsphase des Lebenszyklus. Eine Ausnahme bilden Variantenrelationen. Ereignisse, die zur Entstehung einer Variantenrelation führen, können auch in der Bereitstellungs- bzw. Zugriffsphase auftreten, wenn z.B. ein Dokument im Dateisystem kopiert und umbenannt wird. Es ist daher notwendig, die Aktionen des Nutzers auch in diesen Phasen zu verfolgen.

Zu 2): Eine vollständig beschriebene Relation setzt, abhängig vom Relationstyp, die Erfassung verschiedener Informationen voraus. Als vollständig beschriebene Relation wird eine Relation bezeichnet, die folgende Kriterien erfüllt: Sie besitzt ein eindeutig identifizierbares Quell- und Zieldokument sowie einen Relationstyp. Bei Element- und Aggregationsrelationen müssen die an der Relation beteiligten Elemente innerhalb der Dokumente eindeutig identifizierbar sein. Es müssen also in diesen Fällen zusätzlich Informationen über die Elemente, wie z.B. die ID oder der Typ des Elements, gewonnen werden.

Darüber hinaus ist auch die Erfassung von Verwendungsinformationen in Form von Sessioninformationen zu jeder Relation notwendig. Zu diesen zählen der Nutzer, der die Relation erzeugt hat, und ein Timestamp des Zeitpunkts der Entstehung der Relation. Zusätzlich müssen Metadaten über die an der Relation beteiligten Dokumente wie zum Beispiel Autor, Computername des Rechners, auf dem das Dokument verarbeitet wird, Typ des Dateisystems, in welchem das Dokument gespeichert wird, sowie Angaben über Dateinamen und Pfade erfasst werden.

Zu 3): Es genügt nicht, alle Aktionen eines Nutzers zu verfolgen, die zur Entstehung einer Relation führen können, es muss auch eine Möglichkeit geben, zu erkennen, wenn eine Relation ihre Gültigkeit verliert. Eine Relation kann ihre Validität zum Beispiel dadurch verlieren, dass der Nutzer das entsprechende Element wieder aus dem Dokument oder sogar das Dokument selbst löscht. Bei der Wiederver-

Tabelle 4.1.: Für Erfassung und Verfolgung relevante Aktionen

	Entsteht durch...	**Verliert Validität durch...**
Varianten-relation	- Speichern eines Dokuments mit neuem Dateinamen - Duplizieren des Dokuments außerhalb der Applikation (z.B. im Dateisystem)	- Löschen des Quell- oder Zieldokuments
Element-relation	- Kopieren eines Artefakts in einem Dokument und Einfügen in ein anderes Dokument	- Löschen des Artefakts - Löschen aller Inhalte des Artefakts - Löschen des Quell- oder Zieldokuments - Undo- & Redo-Aktionen
Aggregations-relation	- Kopieren eines Dokuments (z.B. im Dateisystem) und Einfügen in ein anderes Dokument	- Löschen des eingefügten Dokuments - Löschen des Zieldokuments - Undo- & Redo-Aktionen
Verknüpfungs-relation	- Verknüpfen eines Dokuments mit einem anderen Dokument - Kopieren und Einfügen einer URL	- Löschen der Verknüpfung - Löschen des verknüpften Dokuments (Dead Link) - Löschen des Zieldokuments - Undo- & Redo-Aktionen

wendung von Text kommt es weiterhin sehr häufig vor, dass dieser verändert, gekürzt oder erweitert wird. Hier gilt es zu entscheiden, wann eine Relation, die durch die Wiederverwendung dieses Textes entstanden ist, bei dessen Veränderung ihre Gültigkeit verliert. Tabelle 4.1 zeigt eine vollständige Übersicht über die Aktionen und Ereignisse, die zur Folge haben, dass eine Relation ihre Validität verliert. Um zu gewährleisten, dass erfasste Relationen konsistent mit dem Zustand der Dokumente sind, gibt es prinzipiell zwei Ansätze: Die komplette Nachverfolgung der relevanten Aktionen (3a) oder eine regelmäßige Validierung der gesammelten Informationen (3b). Beide werden in Abschnitt 4.1.1 beschrieben.

Zu 4): Die zu einer Relation erfassten Informationen müssen an Änderungen, die außerhalb der überwachten Applikation erfolgen, angepasst werden. Wenn beispielsweise der Titel oder Pfad eines an einer Relation beteiligten Dokuments im Dateisystem, also außerhalb der überwachten Applikation, geändert wird, muss dies reflektiert werden.

Zu 5): Dieser Punkt hängt eng mit Punkt 3 zusammen. Der hier vorgeschlagene Ansatz beruht auf der Erfassung von Informationen während des Lebenszyklus von Dokumenten. Dies geschieht mit Hilfe von Erweiterungen und Plug-ins der Applikationen und Systeme, die zur Verarbeitung der Dokumente genutzt werden. Eine Erfassung setzt hierbei wiederum voraus, dass diese Erweiterungen auf dem Rechner des jeweiligen Nutzers installiert sind. In einem nicht geschlossenen Szenario kommt es jedoch vor, dass Nutzer mit den Dokumenten arbeiten, welche die entsprechenden Erweiterungen nicht installiert haben. Änderungen, die durch diese Nutzer vorgenommen werden, können nicht verfolgt werden und erfasste Relationen ihre Gültigkeit verlieren. Für einen solchen Fall müssen Mechanismen vorgesehen werden, die eine Überprüfung vorhandener Relationen auf ihre Gültigkeit ermöglichen. Solche Mechanismen werden in den Kapiteln 5 und 6 beschrieben.

Abbildung 4.2 fasst die identifizierten Herausforderungen in einem exemplarischen Prozess zusammen. Die Schritte sind entsprechend der beschriebenen Herausforderungen nummeriert. Zunächst wer-

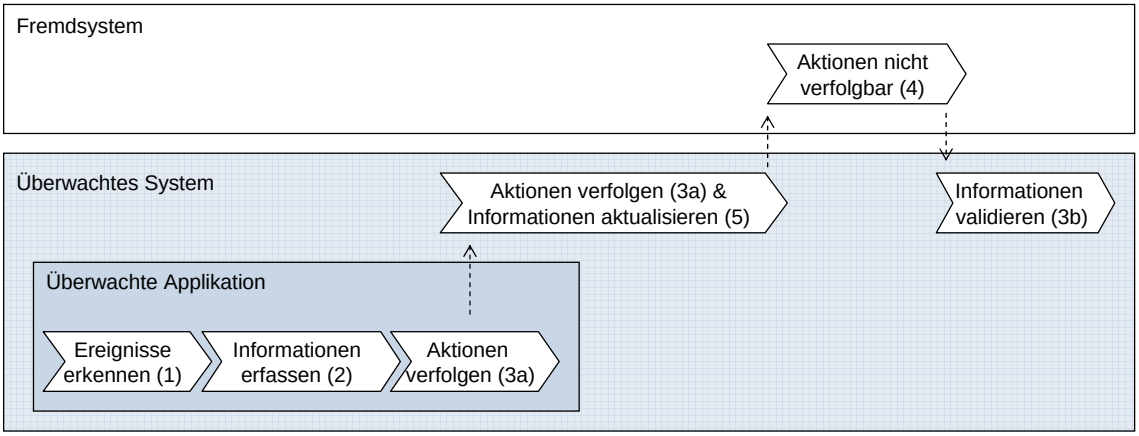

Abbildung 4.2.: Herausforderungen der Erfassung

den die Ereignisse, die zur Entstehung einer Relation führen, erkannt und die notwendigen Informationen erfasst. Nun müssen Aktionen, die zum Validitätsverlust einer Relation führen können, verfolgt werden. Wenn ein solches Ereignis eintritt, müssen entsprechende Maßnahmen ergriffen werden. Auch relevante Aktionen außerhalb der Applikation selbst müssen verfolgt werden. Gegebenenfalls werden die Informationen entsprechend aktualisiert. Schließlich erfordert das System eine Berücksichtigung von Änderungen auf nicht überwachten Systemen. Auf einem solchen System durchgeführte Aktionen, die sich auf die Validität einer Relation auswirken, müssen im Nachhinein erkannt werden, zum Beispiel durch Validierung. Wiederum müssen die erfassten Informationen entsprechend angepasst werden.

Konzepte zur Erhaltung der Konsistenz von Beziehungsinformationen

Wie zuvor beschrieben, genügt es nicht, Relationen zu erfassen, indem man die entsprechenden Ereignisse erkennt und alle notwendige Informationen gewinnt; die erfassten Relationen müssen auch dem wirklichen Zustand der beteiligten Dokumente entsprechen, der sich im weiteren Verlauf des Lebenszyklus der Dokumente verändern kann. Wenn beispielsweise eine Folie aus einer Präsentation in eine andere kopiert wird, entsteht zwischen beiden Präsentationen eine Relation. Wird die kopierte Folie jedoch wieder gelöscht, verliert auch die zugehörige Relation ihre Gültigkeit. Dies muss berücksichtigt und adressiert werden. Dafür wurden zwei unterschiedliche Konzepte entwickelt, die im Folgenden beschrieben werden.

Komplette Nachverfolgung: Bei diesem Konzept werden nicht nur die Ereignisse verfolgt, die zur Entstehung einer Relation führen, sondern auch alle Ereignisse, die dazu führen können, dass eine Relation ihre Validität verliert. Hierzu zählen, wie in Tabelle 4.1 gezeigt, unter anderem das Löschen von Elementen, das Verwerfen von Änderungen und das Rückgängigmachen von Vorgängen, die zur Entstehung einer Relation geführt haben ("Undo"). In solchen Fällen muss die erfasste Relation wieder entfernt werden. Ebenso wie eine erfasste Relation durch diese Aktionen inkonsistent werden kann, kann es vorkommen, dass eine Relation ihre Validität zurückerhält. Dies geschieht zum Beispiel, wenn ein Löschvorgang an einem an einer Relation beteiligten Element rückgängig gemacht wird. Welche dieser Ereignisse

relevant sind, ist vom Relationstyp abhängig. Eine Variantenrelation verliert ihre Validität, wenn zum Beispiel das Quell- oder Zieldokument der Relation gelöscht wird. Bei Element- und Aggregationsrelationen geschieht dies unter anderem, wenn das Element, auf das die Relation zeigt, gelöscht wird oder dessen Einfügen rückgängig gemacht wird. Bei Verknüpfungsrelationen gilt dasselbe in Bezug auf das Einfügen und Löschen der Verknüpfung. Alle zuvor genannten Ereignisse müssen kontinuierlich verfolgt werden, damit dieses Konzept angewendet werden kann. Sobald ein relevantes Ereignis nicht verfolgbar ist, weil beispielsweise die Schnittstelle der entsprechenden Applikation dies nicht zulässt, kann der Ansatz nicht angewendet werden. Ein weiteres Problem besteht darin, dass ein Fehler, der einmal bei der Nachverfolgung aufgetreten ist, immer weiter getragen wird und fast zwangsläufig dazu führt, dass erfasste Informationen fehlerhaft sind.

Regelmäßige Validierung: Oft ist eine komplette Nachverfolgung aller Ereignisse, die zur Ungültigkeit einer Relation führen können, nicht möglich. Dennoch muss dafür gesorgt werden, dass die erfassten Relationen valide und konsistent mit dem Zustand der Dokumente sind. Alternativ zur kompletten Nachverfolgung können die erfassten Relationen auch regelmäßig validiert werden. Dies bedeutet, dass eine automatisierte Untersuchung einer Relation daraufhin stattfindet, ob sie noch gültig ist. Insbesondere bei textlastigen Dokumenten ist diese Methode oft unerlässlich, da Änderungen an Texten nur sehr schwer über die Schnittstelle einer Applikation nachverfolgt werden können. Den Verfahren und Methoden zur Validierung von Relationen sind in dieser Arbeit zwei eigene Kapitel gewidmet, in welchen diese beschrieben und evaluiert werden und ihr Einsatz im Rahmen dieser Arbeit erläutert wird (Kapitel 5 und 6). Auch hier ist das Verfahren, welches zur Validierung angewendet werden kann, wiederum vom Typ der Relation abhängig.

Es ist auch möglich ein Mischverfahren anzuwenden, in welchem die vollständige Nachverfolgung durch eine spätere Validierung abgesichert wird, oder bestimmte Relationstypen nachverfolgt und andere validiert werden.

4.1.2 Erfassung von Verwendungsinformationen

Auch wenn die Erfassung von Verwendungsinformationen nicht im Fokus dieser Arbeit steht, werden die Herausforderungen bei der Erfassung solcher Informationen dennoch im Folgenden analysiert. Wie in Kapitel 2.4.2 gezeigt, ist die Menge der Verwendungsinformationen, die erfasst werden können, sehr groß und sehr heterogen. Daraus ergeben sich zwei prinzipielle Herausforderungen:

1. Nutzbarkeit: Die erfassten Informationen müssen nutzbar sein. Das bedeutet einerseits, dass sie auf eine Art und Weise erfasst (und verwaltet) werden, dass sie genutzt werden können, andererseits aber auch, dass Informationen, die erfasst werden sollen, einen klaren Nutzen haben sollten. Verwendungsinformationen zu erfassen, für die es kein Anwendungsszenario gibt, wäre Ressourcenverschwendung.

2. Heterogenität: Je unterschiedlicher die Informationen, desto unterschiedlicher müssen auch die Methoden sein, sie zu gewinnen. Dementsprechend muss die Erfassung auf unterschiedliche Arten und Quellen von Informationen angepasst werden.

3. Strukturiertheit: Wenn Verwendungsinformationen in einem klar strukturierten Format erfasst werden, sind sie leichter zu verarbeiten. Insbesondere bei der Erfassung von Verwendungsinformationen besteht die Gefahr, dass die Erfassung der Informationen eher einem unstrukturierten Logging ähnelt und damit die Nutzung der Informationen zu einer schwierigen Herausforderung wird.

Verwendungsinformationen sollen unterstützend zu erfassten Beziehungsinformationen für die Nutzung eingesetzt werden. Hierzu zählen beispielsweise Informationen zu Prozessen, die Relationen zur Folge haben, wie z.B. Sessioninformationen. So ist es hilfreich, zu jeder Relation zusätzliche Informationen, wie den Erzeuger, Zeitpunkt, Informationen über den Computer, aber auch über die beteiligten Dokumente zu erfassen. Des Weiteren gibt es viele Arten von Verwendungsinformationen, die in bestimmten Anwendungsszenarien in Verbindung mit Beziehungsinformationen hilfreich sein können. Beispielsweise ist die Anzahl, wie oft ein Dokument angeklickt, angesehen oder heruntergeladen wurde, ein Kriterium, das in Verbindung mit Beziehungsinformationen als Eingabe für Such- und Ranking-Algorithmen verwendet werden kann.

Die Art und Weise, wie Verwendungsinformationen erfasst werden können, hängt davon ab, um welche Informationen es sich handelt. Sessioninformationen können sehr häufig über die Schnittstellen des Betriebssystems abgefragt werden. Dies ist auch bei vielen Umgebungsinformationen der Fall. Applikationsinformationen jedoch erfordern, wie Beziehungsinformationen, zumeist ein Eindringen in die Applikation, in der das Dokument verarbeitet wird. Oft bieten solche Applikationen Schnittstellen an, die eine Erfassung solcher Informationen ermöglichen.

4.2 Verwaltung von Lebenszyklusinformationen

Über die Erfassung von Lebenszyklusinformationen hinaus muss auch eine Speicherung und Verwaltung dieser Informationen erfolgen. Im folgenden Abschnitt werden zunächst die Herausforderungen bei der Verwaltung von Lebenszyklusinformationen aufgezeigt. Es folgt eine Analyse der beiden für die Verwaltung in Betracht kommenden Paradigmen - eine zentralisierte Lösung oder ein verteilter Ansatz. Schließlich wird ein Schema für die strukturierte Speicherung von Beziehungsinformationen vorgestellt.

4.2.1 Herausforderungen

Lebenszyklusinformationen werden zum großen Teil innerhalb der Applikation, in der die zugehörigen Prozesse stattfinden, erfasst. Eine Nutzung dieser Informationen erfolgt jedoch nicht notwendigerweise innerhalb der Applikation, in der sie erfasst wurden. Des Weiteren wird ein Dokument während seines Lebenszyklus oftmals mit verschiedenen Applikationen geöffnet, verwendet oder bearbeitet. Es muss daher dafür gesorgt werden, dass Lebenszyklusinformationen applikationsunabhängig gespeichert werden können. Weiterhin sollen Lebenszyklusinformationen auch benutzerübergreifend erfasst werden. Wenn ein Dokument von verschiedenen Nutzern verwendet wird, muss ein System zur Verwaltung von Lebenszyklusinformationen gewährleisten, dass diese systemübergreifend verwaltet werden können. Dies hat zur Folge, dass die Informationen zwischen den Systemen verschiedener Nutzer ausgetauscht und synchronisiert werden müssen. Auch ein applikationsunabhängiges Speichern von Informationen hat oft zur Folge, dass Systemgrenzen überschritten werden müssen. Wenn ein Dokument beispielsweise in einem

zentralen Repository oder in einem Dokumentenmanagementsystem verwaltet wird, müssen auch die Informationen, die innerhalb dieser Applikationen entstehen, erfasst und verwaltet werden. Die in den unterschiedlichen Applikationen auf unterschiedlichen Systemen erfassten Informationen müssen also so gespeichert werden, dass von allen beteiligten Systemen auf sie zugegriffen werden kann, außerdem muss das System zur Erfassung der Informationen von allen beteiligten Systemen aus erreichbar sein.

Weiterhin wird für die Verwaltung von Lebenszyklusinformationen ein Schema benötigt, welches festlegt, welche Informationen in welchem Format erfasst werden. Ein solches Schema gewährleistet die Austauschbarkeit von Lebenszyklusinformationen zwischen Applikationen, Nutzern und Systemen.

Eine dritte Herausforderung ergibt sich aus der Menge der erfassten Informationen. Es ist zu erwarten, dass die Menge der zu erfassenden Informationen, insbesondere die der Verwendungsinformationen, aber auch die der Beziehungsinformationen, groß ist, besonders wenn die Erfassung der Informationen bei vielen verschiedenen Nutzern erfolgt. Deshalb muss die Verwaltung effizient sein und Mechanismen vorsehen, um beispielsweise redundante Informationen zu entfernen oder die Informationsmenge mit Hilfe von z.B. Kompressionsverfahren zu verkleinern.

Im Folgenden wird zunächst erläutert, wie eine system- und applikationsübergreifende Verwaltung von Lebenszyklusinformationen erfolgen kann. Hierfür werden zwei grundlegende Ansätze erarbeitet und verglichen.

4.2.2 Analyse grundlegender Konzepte

Für eine system- und applikationsübergreifende Verwaltung von Lebenszyklusinformationen gibt es grundsätzlich zwei verschiedene Ansätze: einen zentralisierten oder einen verteilten Ansatz. Ein zentralisierter Ansatz wäre beispielsweise eine Client-Server-Lösung, während eine Umsetzung als Peer-to-Peer (P2P) -Netz einen verteilten Ansatz darstellt.

Der zentralisierte Ansatz sieht die Verwaltung der erfassten Lebenszyklusinformationen in einer zentralen Instanz vor. Diese muss von allen Systemen, auf denen eine Erfassung von Informationen erfolgt, erreichbar sein. Ebenso müssen alle Systeme, in denen die gesammelten Informationen genutzt werden sollen, eine Verbindung zu dieser Instanz aufbauen können. Eine solche Architektur entspricht einem Client-Server-Modell. Der zentrale Server muss dabei entsprechende Schnittstellen zum Abrufen oder Übertragen der Informationen durch die beteiligten Clients anbieten. Der Server dient in diesem Modell also als zentraler Sammelpunkt für alle erfassten Informationen. Dieser Umstand kann genutzt werden, um beispielsweise redundante Informationen zu entfernen oder vorhandene Informationen anzureichern. Abbildung 4.3 zeigt eine schematische Ansicht des zentralisierten Ansatzes.

Die zweite Möglichkeit für die Verwaltung von Lebenszyklusinformationen bietet ein verteilter Ansatz auf Peer-to-Peer-Basis (P2P) (vergleiche Abbildung 4.4). Hierbei werden die Informationen nicht an einer zentralen Stelle gesammelt, sondern liegen auf den Peers des Systems selbst. Ein Overlay-Netzwerk, das die Peers organisiert, ermöglicht hierbei die Verwaltung von Informationen. P2P-Netze sind bekannt für ihre Verwendung als Basis von Dateitauschbörsen wie Napster, Gnutella oder Bittorrent [SE05]. In einer solchen Tauschbörse wird die Suche nach Dateien im P2P-Netz durchgeführt, während die Übertragung der Datei selbst direkt zwischen den am Tausch beteiligten Systemen vonstatten geht. Um ein Auffinden von gewünschten Dateien zu ermöglichen, muss das P2P-Netz demnach die für eine Suche relevanten

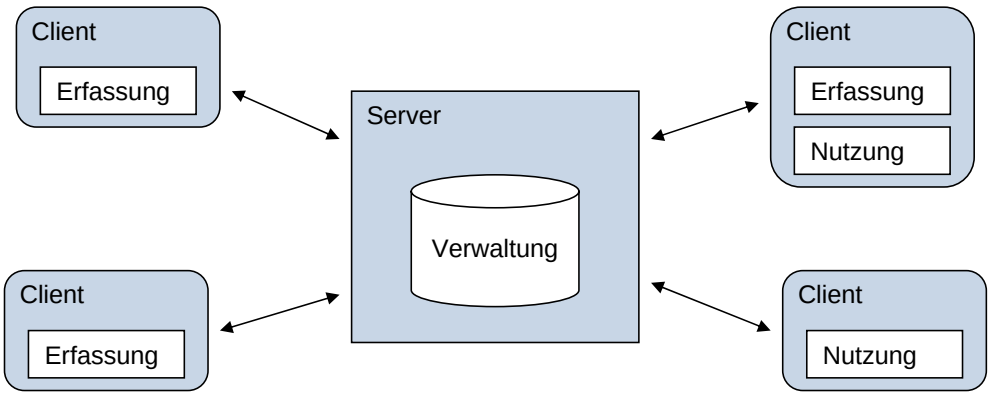

Abbildung 4.3.: Zentralisierter Ansatz

Metadaten der Dateien verwalten und auf Suchanfragen entsprechend reagieren. Ein solches System kann auch modifiziert werden, zum Beispiel, um statt der vorgesehenen Metadaten Lebenszyklusinformationen zu verwalten. In [NWQ+02] wird das P2P-System "Edutella" beschrieben, das der Verwaltung und dem Austausch von mit LOM-Metadaten beschriebenen Lernobjekten dient.

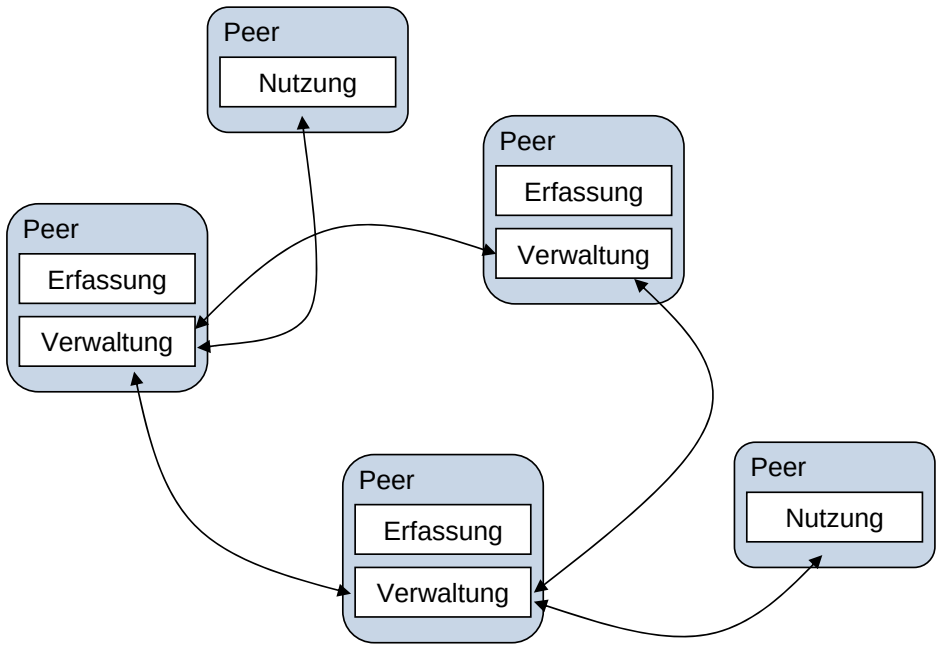

Abbildung 4.4.: Verteilter Ansatz

Sowohl der zentralisierte als auch der verteilte Ansatz bringen verschiedene Vor- und Nachteile mit sich, die im Folgenden erörtert werden.

Verfügbarkeit der Informationen: Bei einer zentralisierten Lösung gibt es *eine* Instanz - den Server, bei dem alle Informationen zusammenlaufen. Dies hat den Vorteil, dass die Daten gesammelt verarbeitet werden können. Dies hat jedoch auch den Nachteil, dass der Server dadurch zur kritischen Komponente

wird, die immer erreichbar sein muss - der sogenannte "single point of failure". Fällt er aus, funktioniert das gesamte System nicht mehr. Dem muss dann mit verhältnismäßig aufwändigen Maßnahmen wie Redundanz entgegengewirkt werden. In einem verteilten System sind die Informationen auf verschiedene Rechner (Peers) verteilt. Diese können dem P2P-Netz nach Belieben beitreten und es auch wieder verlassen. Dieses Problem wird "churn" genannt (siehe [RGRK04], [SR06]). Dadurch ist unmöglich zu gewährleisten, dass immer alle Informationen im Netz vorhanden sind. Selbst wenn dies der Fall wäre, sind sie nicht an einer Stelle gesammelt, sondern auf diverse Peers verteilt. Dies macht eine Verarbeitung und Aufbereitung der Informationen zu einer interessanten, aber sehr komplexen Herausforderung.

Skalierbarkeit: Ein weiteres Entscheidungskriterium ist die Skalierbarkeit des Systems. Wie beschrieben, ist zu erwarten, dass viele Lebenszyklusinformationen anfallen und verwaltet werden müssen, insbesondere bei einer großen Anzahl von Nutzern. Eine P2P-Lösung hat den Vorteil, dass sie besser skaliert als eine zentralisierte Lösung, da die Daten auf diverse Peers verteilt sind. Ein Server ist irgendwann ausgelastet und muss entsprechend erweitert werden. Die zentralisierte Lösung birgt dabei auch die Gefahr, dass der Server zum Flaschenhals wird.

Umsetzbarkeit: Die beiden Ansätze sind in verschiedenen Szenarien unterschiedlich gut umsetzbar. Ein P2P-Ansatz passt prinzipiell gut zu Wissensdokumenten, deren Instanzen häufig verteilt auf den Rechnern unterschiedlicher Nutzer vorliegen und die nicht grundsätzlich in zentralisierten Systemen verwaltet werden, wie es z.B. bei Webseiten der Fall ist. Andererseits wird in einem P2P-Netzwerk zum Austausch von Inhalten oder Informationen eine kritische Masse an Peers benötigt [KWIK03] - dies ist auch bei Lebenszyklusinformationen der Fall. Über die in Kapitel 4.2.1 genannten Herausforderungen hinaus, wirft eine P2P-Lösung weitere komplexe Fragestellungen auf, deren Beantwortung jedoch nicht das Kernziel dieser Arbeit ist. Dazu zählen:

- Was passiert mit Informationen, die sich auf Peers befinden, die gerade offline sind?
- Wie können Informationen im P2P-Netz persistent zugreifbar gemacht werden?
- Wie können die gesammelten Informationen aufbereitet und verarbeitet werden?

Eine Implementierung als zentralisierter Ansatz ist hingegen vergleichsweise einfach umsetzbar.

Beide beschriebenen Ansätze haben sowohl Vor- als auch Nachteile. Beim verteilten Ansatz überwiegen die Nachteile jedoch. Die Umsetzung der Verwaltung von Lebenszyklusinformationen in einem P2P-Netz stellt interessante Herausforderungen. Diesen zu begegnen ist jedoch kein Ziel dieser Arbeit. Die Umsetzung der zentralisierten Lösung ist weniger komplex und gewährleistet die Verarbeitbarkeit aller gesammelten Informationen. Das System sollte jedoch so konzipiert sein, dass kein Informationsverlust auftritt und das System nutzbar bleibt, auch wenn der zentrale Server zeitweise ausfällt oder nicht erreichbar ist.

4.2.3 Speicherung von Lebenszyklusinformationen

Für die Speicherung von Lebenszyklusinformationen wird ein Format benötigt, in welchem die erfassten Informationen vorgehalten werden können. Im Folgenden werden verschiedene bereits bestehende Metadaten-Standards und -Formate untersucht und auf ihre Nutzbarkeit zur Verwaltung von doku-

mentenzentrierten Lebenszyklusinformationen hin beurteilt. Schließlich wird ein Schema zur Speicherung dieser Informationen vorgestellt.

Bestehende Standards

Ein weit verbreiteter Standard zur Beschreibung von Dokumenten, insbesondere im Bereich digitaler Bibliotheken, ist **Dublin Core** [DCM08]. Dublin Core sieht Felder zur Beschreibung des Inhalts eines Dokumentes (zum Beispiel "Titel" oder "Beschreibung"), technischer Eigenschaften (z.B. "Typ", "Format" oder "Sprache") oder der Rechte an einem Dokument ("Autor", "Rechteinhaber") vor. Weiterhin gibt es ein Element ("Relation"), mit dessen Hilfe die Beziehungen des beschriebenen Dokuments zu anderen Dokumenten dargestellt werden können. Es besitzt zwei untergeordnete Elemente: Die ID des Dokuments (oder dessen Metadatensatzes), zu dem die Beziehung besteht, und den Typ der Beziehung. Für Letzteres ist ein bestimmtes Vokabular vorgesehen (siehe [DCM97]), das folgende Relationstypen unterstützt:

- IsPartOf / HasPart: Eine Aggregationsbeziehung, bei der ein Dokument physikalisch der Teil eines anderen ist.
- IsVersionOf / HasVersion: Eine Version eines Dokuments ist eine Instanz in der Versionshistorie des Dokuments, die vom selben Autor z.B. durch erneutes Bearbeiten erzeugt wurde.
- IsFormatOf / HasFormat: Eine solche Beziehung entsteht durch Überführung eines Dokuments in ein anderes Format.
- References / IsReferencedBy: Eine solche Beziehung entsteht durch Zitieren eines anderen Dokuments.
- IsBasedOn / IsBasisFor: Diese Relation entsteht, wenn ein Dokument z.B. übersetzt, produziert, angepasst oder interpretiert wird.
- Requires / IsRequiredBy: Abhängigkeitsrelationen entstehen, wenn ein Dokument beispielsweise zum Verständnis oder zur Nutzung eines anderen Dokuments benötigt wird.

Dieses Vokabular lässt sich nicht unmittelbar auf die in 2.4.3 beschriebenen Beziehungstypen anwenden. Während Aggregationsrelationen mit "IsPartOf/HasPart" abbildbar sind, entsprechen Variantenrelationen einer Mischung aus den Dublin-Core-Relationen "IsVersionOf/HasVersion" und "IsBasedOn/IsBasisFor". Verknüpfungsrelationen sind zwar mit den in Dublin-Core genannten Referenzen verwandt, haben aber nicht dieselbe Bedeutung. Elementrelationen werden von Dublin Core überhaupt nicht abgedeckt. Zudem fehlt die Möglichkeit, weitere Informationen über Relationen, wie zum Beispiel Informationen über die in Beziehung stehenden Elemente, festzuhalten.

Ein weiteres Problem ist die Verbreitung des Standards im Anwendungsfeld der Wissensdokumente. Wissensdokumente sind im Allgemeinen nicht mit Dublin-Core-Metadaten beschrieben. Um diesen Standard für die Verwaltung von Lebenszyklusinformationen verwenden zu können, müsste er stark angepasst werden. Dies ist jedoch nur sinnvoll, wenn er unter den Dokumenten, für die die Informationen erfasst werden sollen, schon weit verbreitet ist. Dies ist hier nicht der Fall.

Im E-Learning Bereich hat sich **LOM** (Learning Object Metadata) [IEE02] als Standard für Metadaten etabliert. Er sieht neun übergeordnete Kategorien vor (zum Beispiel: "General", "Educational", "Technical") mit über 60 Metadatenfeldern vor. Ein LOM-Metadatensatz bezieht sich grundsätzlich auf ein Lernobjekt. In einer der neun Kategorien ("Relation") ist das Ablegen von Informationen über zu dem entsprechenden Lernobjekt in Beziehung stehende Objekte vorgesehen. In dieser Kategorie wird neben dem Relationstyp auch die Ressource, mit der das Lernobjekt in Beziehung steht, gespeichert. Als Vokabular für die Relationstypen wird das oben beschriebene Vokabular des Dublin-Core-Standards verwendet. Mit Hilfe von Applikationsprofilen können Standards wie LOM für bestimmte Anwendungen angepasst werden, indem zum Beispiel die Wertebereiche bestimmter Felder eingegrenzt, Beziehungen zwischen Feldern definiert oder Namespaces zur Erweiterung des Standards hinzugefügt werden. Durch die Nutzung des Dublin-Core-Vokabulars zur Beschreibung von Relationen ist der LOM-Standard in dieser Form ebenso wenig für die Verwaltung von Lebenszyklusinformationen nutzbar wie Dublin Core selbst. Auch die Ausrichtung von LOM auf Lernobjekte spricht gegen eine Verwendung dieses Standards. In Anwendungsszenarien, in denen Lebenszyklusinformationen speziell für Lernobjekte erfasst werden sollen, ist es jedoch sinnvoll, die Möglichkeiten der Erweiterbarkeit des Standards zu nutzen und die für die Verwaltung von Lebenszyklusinformationen notwendigen Felder in den LOM-Standard zu integrieren. [LHRS07] beschreibt einen Ansatz zur Verwaltung von Beziehungsinformationen in LOM durch eine Erweiterung der Relation-Kategorie.

Im Bereich der Langzeitarchivierung von digitalen Objekten gibt es das OASI-Referenzmodell [OAI02], welches auch ein Informationsmodell zur Beschreibung archivierter Objekte beinhaltet. Es ist in diesem Bereich anerkannt und verbreitet. Brocks et al. haben dieses Informationsmodell erweitert, um auch Kontextdaten erfassbar zu machen [BKJH10]. Brocks Definition von Kontextdaten und die hier verwendete Definition von Lebenszyklusinformationen weisen zwar Überlappungen auf, Kontextdaten umfassen neben dokumentenzentrierten Informationen jedoch auch prozesszentrierte Informationen. Zudem ist das OAIS-Modell allein auf Archivierung ausgerichtet und findet bei Dokumenten außerhalb von Archiven so gut wie keine Verwendung. Die Langzeitarchivierung von Wissensdokumenten wird in dieser Arbeit nicht berücksichtigt, weshalb sich dieses Modell als ungeeignet erweist.

Das in Kapitel 3.1.1 beschriebene CAM-Schema bietet ein Format zur Verwaltung von nutzerzentrierten Verwendungsinformationen, sogenannter "Attention Metadata". Die Aktionen eines Nutzers in verschiedenen Applikationen auf verschiedenen Dokumenten werden mit Hilfe von Feeds gespeichert (siehe Kapitel 3). Das Format sieht die Erfassung von Applikationsinformationen vor, zudem können für jede erfasste Aktion Sessioninformationen gespeichert werden. Auch die Erfassung von Umgebungsinformationen wird durch das Schema unterstützt. Hinsichtlich der Erfassung von Verwendungsinformationen ist dieses Format vollständig. Weiterhin ist es möglich, die eigentlich als nutzerzentriert konzipierte Erfassung in ein Format zur dokumentenzentrierten Erfassung von Informationen zu überführen. Was das CAM-Schema jedoch nicht vorsieht, ist die Speicherung von Beziehungsinformationen. Daher wird im Folgenden das im Rahmen dieser Arbeit entwickelte LIS.KOM-Schema zur Verwaltung von Beziehungsinformationen vorgestellt und gezeigt, wie dieses in das CAM-Schema integriert werden kann.

Das LIS.KOM-Schema wird in Form eines relationalen Modells umgesetzt, da einerseits diese Art der Repräsentation durch existierende und verfügbare Werkzeuge die beste Unterstützung erfährt und andererseits die Leistungsfähigkeit relationaler Systeme im Vergleich zu beispielsweise objektorientierten Repräsentationen im Hinblick auf eine Nutzung der Informationen höher ist. Zudem ist die Anzahl der zu verwaltenden Informationstypen deterministisch; in diesem Fall ist ein relationales Modell gut geeignet.

Lebenszyklusinformationen sind der Definition nach dokumentenzentriert. Es sind die Informationen, die im Laufe des Lebenszyklus eines Wissensdokuments entstehen. Aus diesem Grund steht der Dokumentknoten im Zentrum des Schemas (vergl. Abbildung 4.5), das heißt, alle Informationen werden über ein bestimmtes Dokument gesammelt. Ein Dokument besitzt hierbei eine eindeutige ID, anhand derer es sich identifizieren lässt. Weiterhin hat es einen Besitzer, also einen Nutzer, dem es zugeordnet ist. Ein Fingerprint (siehe Kapitel 6) - eine kurze Repräsentation des Dokuments - zählt ebenfalls zu den Attributen eines Dokuments, so wie auch der Titel. Unabhängig vom System, in dem ein Dokument abgelegt ist, besitzt seine physikalische Ausprägung einen Speicherort. Informationen über diesen werden unter dem entsprechenden Knoten zusammengefasst. Dazu zählen die IP-Adresse des Rechners, auf dem das Dokument gespeichert ist, der Rechnername sowie der Typ des Systems. Hierbei handelt es sich zumeist um Dateisysteme, wobei der Systemtyp hier auch die Art des Dateisystems (also NTFS, FAT etc.) beinhaltet. Eine externe ID kann notwendig sein, wenn das Dokument in einem System abgelegt ist (z.B. einer Datenbank oder einem Repository), welches über ein eigenes ID-Handling verfügt und die ID für den Zugriff auf ein Dokument benötigt wird.

Beziehungsinformationen für ein Dokument werden wie folgt im Schema abgebildet: Jedes Dokument kann beliebig viele *Relationen* zu anderen Dokumenten besitzen. Jede Relation hingegen ist genau zwei Dokumenten zugeordnet. Einem Quell- und einem Zieldokument. Eine Relation besitzt verschiedene Attribute. Dazu zählen eine eindeutige ID, der Relationstyp (siehe 2.4.3), ein Zeitstempel der Erzeugung der Relation sowie der Nutzer, der für die Erzeugung der Relation verantwortlich ist. Weiterhin sind zwei Felder vorgesehen, in denen die Ergebnisse der Berechnung der Abdeckung von Quell- und Zieldokument, bzw. -artefakt der Relation gespeichert werden (siehe Kapitel 5.1.3 für eine Definition des Abdeckungsmaßes). Diese Felder werden für die Validierung der Relation benötigt.

Eine Relation kann auf ein sogenanntes *Artefakt* verweisen. Dieses bildet, wenn vorhanden, das Element innerhalb des Quell- bzw. Zieldokumentes ab, welches an der Beziehung beteiligt ist. Als Artefakt wird in diesem Zusammenhang also der wiederverwendete Teil eines Dokuments, der die Entstehung einer Relation zur Folge hat, bezeichnet. Je nach Typ der Relation besitzt diese kein, ein oder zwei Artefakte. Besteht beispielsweise eine "istTeilVon"-Beziehung zwischen einem Bild und einem Word-Dokument, so besitzt die Relation *ein* Artefakt, nämlich das Zielartefakt. In diesem kann dann zum Beispiel die Position des wiederverwendeten Bildes innerhalb des Word-Dokuments oder seine ID abgelegt werden. Bei einer Elementrelation, also wenn beispielsweise Text zwischen zwei Word-Dokumenten kopiert wird, besitzt die resultierende Relation *zwei* Artefakte. Eine Variantenrelation (siehe Kapitel 2.4.3) besitzt kein Artefakt. Jedes Artefakt hat einen Typ, der darüber Auskunft gibt, was genau wiederverwendet wurde. In vielen Fällen ist es möglich, das Artefakt anhand einer ID zu identifizieren. Hierfür ist das ID-Attribut vorgesehen. Wenn Text kopiert wird, kann dieser im Allgemeinen nicht anhand einer ID identifiziert

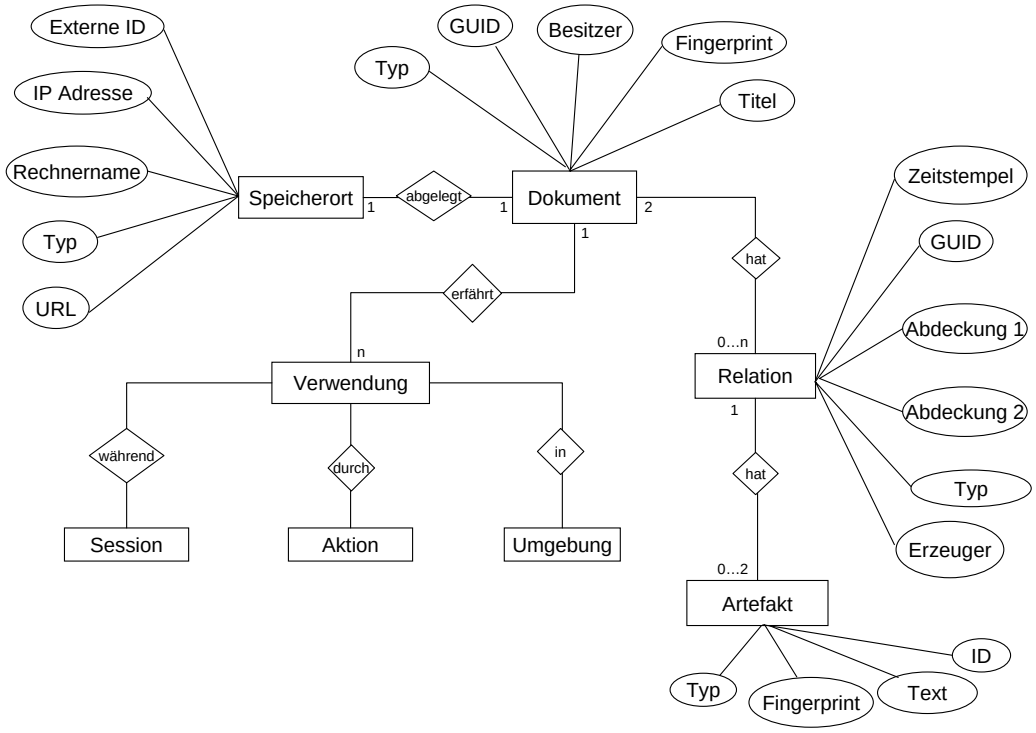

Abbildung 4.5.: Das LIS.KOM-Schema

werden, weshalb in diesen Fällen der kopierte Text selbst gespeichert wird. Dies hilft bei der Validierung der Relation. Weiterhin wird für bestimmte Artefakte ein Fingerprint vorgehalten, welcher ebenso der Validierung und Identifikation dient.

Jedes Dokument erfährt in seinem Lebenszyklus beliebig viele *Verwendungen*. Dementsprechend sind dem Dokument-Knoten Verwendungs-Knoten zugeordnet, mit denen *Verwendungsinformationen* verwaltet werden können. Jede Verwendung besteht aus der entsprechenden Aktion, der Umgebung und der Session. Hier werden die zugehörigen Applikations-, Umgebungs- und Sessioninformationen abgelegt, die der jeweiligen Verwendung zugeordnet sind. Das gewählte Format zur Verwaltung von Verwendungsinformationen ist in das bestehende CAM-Format überführbar. Dieses wird bereits in verschiedenen Applikationen und Ansätzen für die Erfassung und Verwaltung von Verwendungsinformationen genutzt (z.B. [VOD08, WNVD07, TSQ07]). Der Dokument-Knoten in dem hier vorgestellten Schema entspricht dabei dem "Item" aus dem CAM-Schema. Alle Attribute des "Item"-Knotens sind dabei auch beim Dokument-Knoten vorhanden. Eine Verwendung entspricht einem "Event" in CAM und kann auch in dieses überführt werden. Die Aktion entspricht dem CAM-"Action"-Element. Umgebungsinformationen werden in CAM als Kontext bezeichnet, während Sessioninformationen ebenso im Element "Session" abgelegt werden. Einige der im CAM-Schema genutzten Attribute sind für die hier betrachteten Wissensdokumente nicht relevant, dazu zählen zum Beispiel "xnf" oder "followedLinks". Da sie im CAM-Schema auch optional sind, können sie bei einer Überführung weggelassen werden.

Wird das CAM-Schema durch das gegebene LIS.KOM-Schema zur Verwaltung von Beziehungsinformationen erweitert, kann dadurch eine der Schwächen des CAM-Frameworks behoben werden. Während es in der Originalversion des Schemas nicht möglich ist, die Verbindung zwischen zwei identischen Instanzen eines Dokuments bei verschiedenen Nutzern zu erkennen und zu modellieren, kann dies nun

mit Hilfe einer Relation geschehen. Dies eröffnet die Möglichkeit, die gesammelten Attention-Metadaten der beiden Instanzen eines Dokuments zu vereinigen, was einen Informationsgewinn zur Folge hat.

Eine Herausforderung, die sich stellt, ist die Gewährleistung der referentiellen Integrität. Das Schema beinhaltet diverse Referenzen auf ID-Basis, wie zum Beispiel den Bezug zwischen einem Dokument und den zugehörigen Lebenszyklusinformationen, die URL oder ID eines Speicherorts oder die ID eines Artefakts. Bei allen diesen Referenzen besteht die Gefahr, dass sie zur Laufzeit des Systems ungültig werden. Innerhalb des LIS.KOM-Frameworks wird durch diverse Mechanismen sichergestellt, dass Änderungen an Dokumenten, deren Attributen oder deren Speicherort oder Artefakten reflektiert werden (siehe 4.1.1). Wenn ein Dokument den Einflussbereich des LIS.KOM-Frameworks verlässt, ist dies nicht mehr sichergestellt. Es wird jedoch sichergestellt, dass Konflikte, die beim Wiedereintritt von Dokumenten in das LIS.KOM-Framework entstehen, aufgelöst werden (siehe Kapitel 7.3). Wenn Lebenszyklusinformationen aus dem LIS.KOM-Framework bei der Archivierung von Dokumenten berücksichtigt werden sollen, müssen auch hier die diversen Verweise beachtet und Konflikte entsprechend behandelt werden. Die Archivierung von Wissensdokumenten wird in dieser Arbeit jedoch nicht betrachtet.

4.2.4 Verarbeitung von Lebenszyklusinformationen

Das vorgestellte Schema erlaubt eine strukturierte und vollständige Verwaltung von Lebenszyklusinformationen. Damit die gesammelten Informationen in verschiedenen Applikationen so effektiv wie möglich genutzt werden können, ist es sinnvoll, sie darauf aufbauend weiter zu verarbeiten. Hierfür gibt es unterschiedliche Ansätze, die im Folgenden beschrieben werden.

Anreicherung

Eine Anreicherung erfolgt durch Vermehrung vorhandener Information. Im Fall von Beziehungsinformationen können beispielsweise aus bereits bestehenden Beziehungen zwischen Dokumenten weitere geschlossen werden. Ein Beispiel hierfür ist die einfache transitive Folgerung aus einer "istTeilVon"-Relation:

$$A \subseteq B \quad \wedge \quad B \subseteq C \quad \Rightarrow \quad A \subseteq C$$

Wenn Dokument A Teil von Dokument B ist und Dokument B Teil von Dokument C, dann ist auch Dokument A Teil von Dokument C. Bei Wissensdokumenten sind die Hierarchien jedoch meistens eher flach, so dass eine solche Transition selten anwendbar ist. Für Variantenrelationen können jedoch ähnliche Regeln angenommen werden. Zwar kann eine transitive Folgerung in diesem Fall keine verlässliche Aussage treffen, durch sie können jedoch Vorschläge für Anreicherungen generiert werden. Diese können dann mit Hilfe von Validierungstechniken bestätigt oder verworfen werden. Eine weitere Form der Anreicherung ergibt sich ebenfalls durch die Bildung von Variantenrelationen. Da Varianten zum Zeitpunkt der Entstehung der Relation immer ein Duplikat des Quelldokuments sind, können alle Relationen des Quelldokuments auch in die erzeugte Variante übernommen werden. Die übernommenen Relatio-

nen müssen schließlich wiederum nachverfolgt oder zeitweise validiert werden, um ihre Korrektheit zu gewährleisten.

Eine Anreicherung kann auch in der Form erfolgen, dass aus Beziehungsinformationen Metadaten, bzw. Verwendungsinformationen generiert werden. So lässt sich mit Hilfe aller "istTeilVon"- bzw. der "liefertElementAn"-Relationen eines Dokuments bestimmten, wie oft ein Dokument oder Teile daraus wiederverwendet werden. Dies kann auf den gesamten Variantenstrang eines Dokuments ausgeweitet werden. Auch aus Autorensicht können Wiederverwendungsstatistiken generiert werden. So kann ermittelt werden, wie oft ein Autor Inhalte wiederverwendet und ob er Dokumente bestimmter Nutzer öfter verwendet, als die anderer. Hieraus lassen sich Präferenzen ableiten, die dann beispielsweise in Recommender-Systemen oder zur Personalisierung genutzt werden können.

Konsolidierung, Umwandlung & Gewichtung von Relationen

Eine *Konsolidierung* ist vor allem für Beziehungsinformationen sinnvoll. Es geschieht häufig, dass mehrere Artefakte (zum Beispiel Folien) aus einem Dokument in ein und demselben anderen Dokument wiederverwendet werden. Dadurch ergeben sich mehrere Relationen desselben Typs zwischen demselben Quell- und Zieldokument. Für die meisten Anwendungen von Beziehungsinformationen ist eine solche Information redundant. Deshalb können diese Relationen zu einer einzelnen Elementrelation zusammengefasst werden. Da einer solchen Relation mehr Bedeutung zugemessen werden sollte als einer einfachen Elementrelation, kann sie gewichtet werden. Dies kann beispielsweise entsprechend der Anzahl der zusammengefassten Relationen geschehen. Gleiches gilt für Aggregations- und Verknüpfungsrelationen. Wenn ein Bild in einem Dokument mehrfach verwendet wird, ergeben sich wiederum mehrere Relationen mit gleichem Quell- und Zieldokument. Auch diese können zusammengefasst werden. Eine Konsolidierung ist jedoch mit Informationsverlust verbunden, da nicht alle Artefakte der Relationen erhalten bleiben. Deshalb sollten die Originalrelationen nicht mit den konsolidierten Daten überschrieben werden, sondern die Konsolidierung als Vorverarbeitungsschritt vor der Nutzung der Informationen erfolgen.

Auch Variantenrelationen können konsolidiert werden. Wenn beispielsweise ein Dokument, das eine Variante besitzt und selber Variante eines anderen Dokuments ist, gelöscht wird, kann das gelöschte Dokument kurzgeschlossen werden. Eine Gewichtung erfolgt bei Variantenrelationen anhand der Ähnlichkeit der Dokumente. Diese wird mit Validierungstechniken bestimmt (Kapitel 6).

Es kann sinnvoll sein, über eine Konsolidierung von Relationen hinaus, eine Umwandlung der erfassten Relation vorzunehmen. Dies gilt insbesondere für Varianten- und Elementrelationen. Welche Art von Relation erfasst wird, hängt grundsätzlich von der Aktion des Nutzers ab. Wenn ein Nutzer beispielsweise ein bestehendes Dokument unter einem neuen Namen speichert und anpasst, wird eine Variantenrelation zwischen den beiden Ausprägungen des Dokuments erfasst. Kopiert er jedoch große Teile oder den kompletten Inhalt des Dokuments in ein neues Dokument, so entstehen stattdessen viele einzelne Elementrelationen. Es entstehen also unterschiedliche Relationen, obwohl das Ergebnis der Aktion in beiden Fällen dasselbe ist. Um dies zu berücksichtigen, müssen Relationen umgewandelt werden können. Bei oben genanntem Beispiel sollten die Elementrelationen zu einer Variantenrelation zusammengefasst werden, sobald eine bestimmte Anzahl an gemeinsamen Relationen erreicht ist oder die Ähnlichkeit der

Tabelle 4.2.: Überblick über Verarbeitungsmöglichkeiten

	Konsolidierung & Gewichtung	**Umwandlung**
Varianten-relationen	- Kurzschließen von gelöschten Dokumenten im Variantenstrang - Shortcuts im Variantenstrang - Gewichtung von Variantenrelationen anhand der Ähnlichlichkeit	- Umwandlung einer Variantenrelation in diverse Elementrelationen
Element-relationen	- Konsolidierung mehrerer Elementrelationen zwischen denselben Dokumenten zu einer gewichteten Elementrelation	- Umwandlung von vielen Elementrelationen in eine Variantenrelation - Umwandlung einer Elementrelation in eine Verknüpfungsrelation
Aggregations-relationen	- Konsolidierung mehrerer Aggregationsrelationen zwischen denselben Dokumenten zu einer gewichteten Aggregationsrelation	- keine Umwandlung
Verknüpfungs-relationen	- Konsolidierung diverser Verknüpfungsrelationen eines Dokuments mit demselben Verknüpfungsziel zu einer Relation	- keine Umwandlung

Dokumente eine bestimmte Grenze überschreitet. Auch in entgegengesetzte Richtung sollten Relationen angepasst werden können. Eine Variantenrelation setzt einen gewissen Grad an Ähnlichkeit zwischen den beiden verbundenen Dokumenten voraus. Wenn jedoch eins der Dokumente soweit verändert wird, dass die Ähnlichkeit der beiden Dokumente unter eine bestimmte Grenze fällt, verliert die Variantenrelation ihre Validität. In einem solchen Fall sollte die Relation entfernt werden. Dennoch können die Dokumente noch gemeinsame Artefakte besitzen. Für diese werden im Fall einer solchen Umwandlung dann entsprechende Elementrelationen angelegt. Dies ist jedoch nur mit Hilfe der in Kapitel 5 & 6 beschriebenen Ansätze zur Validierung möglich. In beiden Fällen ist es eine Herausforderung, die jeweils günstigen Grenzwerte für eine Umwandlung zu bestimmten. Es kann auch sinnvoll sein, eine Elementrelation in eine Verknüpfungsrelation umzuwandeln. Dies ist dann der Fall, wenn es sich bei dem kopierten Artefakt um einen Link handelt. Tabelle 4.2 zeigt eine Übersicht der Konsolidierungs- und Umwandlungsmöglichkeiten der verschiedenen Relationstypen.

4.3 Nutzung von Lebenszyklusinformationen

Wie die zahlreichen in Kapitel 3.2 beschriebenen Ansätze zur Nutzung von Lebenszyklusinformationen zeigen, sind die Möglichkeiten, Lebenszyklusinformationen nutzbar zu machen, vielfältig. Im Folgenden werden verschiedene Möglichkeiten zur Nutzung von Beziehungsinformationen dargestellt. Sie teilen sich in zwei unterschiedliche Szenarien auf: Die Unterstützung der Auffindung von Dokumenten sowie die Unterstützung von Autorenprozessen. Die Darstellung ist nicht vollständig, da dies nicht im Fokus dieser Arbeit liegt, soll hier aber erfolgen, um einen kompletten Überblick über alle Phasen des Lebenszyklus und alle konzeptionellen Komponenten des Systems zu geben.

4.3.1 Unterstützung des Retrievals von Dokumenten

Eine Möglichkeit der Nutzung, die viele der verwandten Ansätze verfolgen, liegt darin, das Auffinden von Wissensdokumenten zu unterstützen. Dazu gibt es wiederum verschiedene Ansatzpunkte. Beziehungsinformationen können direkten Einfluss auf das Suchergebnis in einem Datei- oder Dokumentenmanagementsystem haben, indem sie als Eingangsdaten für *Ranking-Algorithmen* dienen. So kann eine Abhängigkeit von Wiederverwendung und Beliebtheit eines Dokuments angenommen und Dokumente, die häufig Teil von Wiederverwendungsbeziehungen sind, höher eingestuft werden. Dies wird beispielsweise in [VOD08] oder [OD06] praktiziert. Auch in [LRS07] wird ein Ansatz zum Ranking von Wissensdokumenten mit Hilfe von Beziehungsinformationen vorgestellt. Ochoa stellt in [OD06] einen Ansatz zum Ranking von Lernressourcen mit Hilfe von Verknüpfungen vor. Diese werden jedoch indirekt aus als CAM-Metadaten gespeicherten Interaktionen von Nutzern mit den Lernressourcen gewonnen. Aus diesen Verknüpfungen werden Verbindungen zwischen einzelnen Lernressourcen gewonnen und in das Ranking einbezogen. Ebenso gut, oder sogar noch besser, könnten direkte, als Lebenszyklusinformationen gesammelte Relationen (Beziehungsinformationen) für das Ranking berücksichtigt werden.

Selbiges gilt für die indirekten Beziehungen, die Chirita et al. aus Nutzungssequenzen von Dokumenten ableiten. Auch hier kann gefolgert werden, dass Beziehungsinformationen verlässlichere Verbindungen zwischen Dokumenten darstellen als indirekt aus Aktivitäten geschlossene Verwandtschaften. Ochoa vergleicht seinen vorgeschlagenen Ranking-Algorithmus in [OD06] mit dem Google PageRank-Algorithmus [PBMW99], der für die Websuche verwendet wird. In diesem werden Verknüpfungen sowohl als Verwandtschaft, aber vor allem auch als "Stimme" für eine verlinkte Website gewertet. Ochoa merkt dabei an, dass dies für aus CAM-Metadaten abgeleitete Verbindungen zwischen Dokumenten nicht der Fall sei, weshalb eine direkte Übertragbarkeit nicht gegeben ist. Beziehungsinformationen, wie sie in dem hier vorgestellten System erfasst werden, können jedoch durchaus als "Stimme" für ein Dokument gewertet werden. Somit könnten erfasste Beziehungen sehr einfach in existierende Algorithmen integriert werden.

Auch im Such- und Indexierungs-Framework Lucene [HGM09] können Beziehungsinformationen einfach zur Gewichtung eines Rankings verwendet werden. Das Framework bietet die Möglichkeit, das Ranking eines Dokuments mit Hilfe eines Faktors zu gewichten. So kann zum Beispiel erreicht werden, dass ein Dokument, dessen Artefakte in vielen anderen Dokumenten wiederverwendet werden, höher eingestuft wird als andere Dokumente, die noch nicht wiederverwendet wurden. Hierbei ist es wiederum wichtig zu wissen, welches Dokument das Ziel und welches die Quelle einer Wiederverwendungsbeziehung ist. Diese Information ist im Fall der Erfassung der Beziehung aus dem Lebenszyklus gegeben.

Des weiteren können Beziehungsinformationen einfach genutzt werden, um Suchergebnisse zu erweitern oder Vorschläge zu generieren. Dokumente, zwischen denen eine Wiederverwendungsbeziehung besteht, stehen häufig in einem engen inhaltlichen Zusammenhang. Wenn beispielsweise ein Benutzerhandbuch zu einer Software erstellt werden soll, ist es häufig der Fall, dass Elemente aus bereits bestehenden Dokumenten zu dieser Software wiederverwendet werden. In diesem Fall könnte es sich dabei um die Entwicklerdokumentation, eine bereits bestehende Präsentation oder die FAQ auf der Projektseite der Software handeln. Wenn ein Nutzer die Nutzerdokumentation findet, kann er dadurch auch mit Hilfe

der erfassten Beziehungen die anderen Dokumente auffinden. Wenn also durch Wiederverwendungsbeziehungen in Relation stehende Dokumente für das Suchergebnis berücksichtigt werden, kann das zu umfangreicheren Ergebnislisten führen. Auch ein Browsen entlang der Beziehungen eines Dokuments kann sehr einfach verwirklicht werden. Einen Startpunkt vorausgesetzt, ermöglicht dies ein Auffinden von Dokumenten, ohne dass eine Eingabe von Suchbegriffen erforderlich ist.

4.3.2 Unterstützung von Autorenprozessen

Neben der Unterstützung des Retrievals können Beziehungsinformationen auch dafür genutzt werden, Nutzer bei der Erstellung und Überarbeitung von Dokumenten zu unterstützen. In [MRS07] wird eine Methode zur Unterstützung von Authoring-by-Aggregation beschrieben. Dabei werden Suchanfragen mit Hilfe von aus dem Aggregations-Kontext des bearbeiteten Lernobjekts gewonnenen Informationen spezifiziert. Ähnliches ist auch für Lebenszyklusinformationen denkbar. Es können beispielsweise ausschließlich zu einem Dokument in Beziehung stehende Dokumente in eine Suche einbezogen werden oder Dokumente, die in einer bestimmten Umgebung mit dem aktuellen Dokument verwendet wurden.

Eine zweite Möglichkeit stellt die kontextsensitive Anzeige von Beziehungsinformationen dar. Es kann bei der Bearbeitung eigener Dokumente wie auch der Dokumente anderer Autoren hilfreich sein, zu wissen, mit welchen Dokumenten das bearbeitete Dokument in welcher Weise in Beziehung steht. Insbesondere die Information, welche Dokumente die Quellen eines fertigen Dokuments darstellten, kann für einen fremden Autor nützlich sein. Auch beim kollaborativen Arbeiten an einem Wissensdokument ist es für einen Autor hilfreich, zu sehen, welche Teile des Dokuments aus welchen Quellen stammen. Diese können dann mit Hilfe der erfassten Beziehungsinformationen visualisiert und direkt aus dem Bearbeitungskontext heraus geöffnet werden.

Eine dritte Möglichkeit, die Informationen zur Unterstützung des Autors eines Dokuments zu nutzen, bieten Notifikationen. Ein Autor kann ein Dokument registrieren, um notifiziert zu werden, wenn das Dokument oder eine Instanz des Dokuments verändert wird oder Teile des Dokuments wiederverwendet werden. Oftmals stellen Autoren ihre Dokumente anderen zur Verfügung, ohne Informationen darüber zu erhalten, was mit den Dokumenten geschieht und wer sie in welcher Form wiederverwendet oder überarbeitet. Meist hat der Autor aber ein Interesse daran, dies zu erfahren und gegebenenfalls Zugriff auf überarbeitete oder korrigierte Versionen seiner Dokumente zu erhalten.

Auch in die andere Richtung können Notifikationen genutzt werden. Wenn man selber ein Dokument wiederverwendet, kann man dieses registrieren, um notifiziert zu werden, wenn der Autor des Dokuments dieses verändert.

In einem Archivierungsszenarium ist es häufig notwendig, zu wissen, wer mit einem Dokument wie interagiert hat und wer an der Entstehung und Bearbeitung eines Dokument beteiligt war. Die Erfassung von Lebenszyklusinformationen kann dabei helfen, solche Provenance-Daten für archivierte Dokumente zu generieren.

4.4 Zusammenfassung

In dem vorliegenden Kapitel wurden die Herausforderungen bei der Erfassung und Verwaltung von Lebenszyklusinformationen analysiert und verschiedene Konzepte für die Erfassung von Lebenszyklusinformationen und die Gewährleistung ihrer Gültigkeit erarbeitet. Es wurde ein Schema für die Verwaltung von Beziehungsinformationen vorgestellt und verschiedene Verarbeitungsmethoden für erfasste Lebenszyklusinformationen erläutert. Darüber hinaus wurden Konzepte für verschiedene Nutzungsszenarien entworfen. Zusammenfassend ergeben sich für Erfassung, Verwaltung und Nutzung von Lebenszyklusinformationen folgende Vorgaben:

- Die Erfassung der Informationen muss innerhalb der Applikation erfolgen, in denen die Informationen entstehen. Dies erfordert Plug-ins.
- Die Entwicklung verschiedener Plug-ins ist aufwändig, deshalb sollten alle Komponenten, welche die verschiedenen Plug-ins gemeinsam haben, ausgelagert werden.
- Es muss eine Validierung der erfassten Informationen erfolgen, da die komplette Nachverfolgung der erfassten Informationen in vielen Fällen nicht umsetzbar ist.
- Eine zentralisierte Architektur ist einem P2P-Ansatz vorzuziehen. Die Grundfunktionalität des Systems sollte jedoch erhalten bleiben, auch wenn der Server nicht erreichbar ist.
- Die Nutzung der Informationen kann wiederum innerhalb eines Plug-ins erfolgen, alternativ kann jedoch auch eine eigene Applikation für die Nutzung von Informationen entwickelt oder eine bestehende erweitert werden.

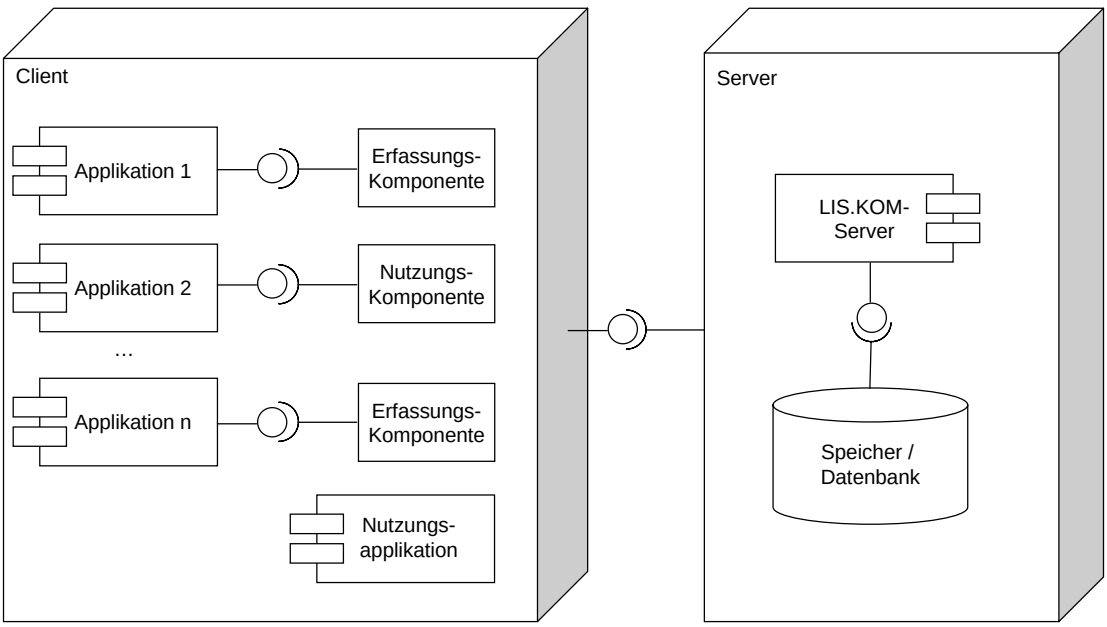

Abbildung 4.6.: Komponenten des LIS.KOM-Frameworks

Anhand dieser Vorgaben kann ein abstraktes Diagramm für ein System zur Erfassung, Verwaltung und Nutzung von Lebenszyklusinformationen entwickelt werden. Abbildung 4.6 zeigt die Komponenten des konzipierten *LIS.KOM-Frameworks*. Die Erfassung erfolgt wie beschrieben innerhalb der verschiedenen Applikationen, die Teil des Lebenszyklus von Wissensdokumenten sein können. Hierzu zählen vor allem Autorenwerkzeuge, Office-Applikationen, Dateisysteme, Repositories oder Content- und Dokumentenmanagement-Systeme. Die gesammelten Informationen werden an den zentralen Server gesendet. Die Nutzung kann in Plug-ins oder stand-alone Applikationen erfolgen.

Teil III.

Validierung von Lebenszyklusinformationen

5 Validierung von Elementrelationen mit Hilfe von String-Matching-Techniken

In diesem und dem folgenden Kapitel werden Ansätze beschrieben, mit deren Hilfe Wiederverwendungsbeziehungen zwischen Dokumenten auf Inhaltsebene validiert werden können. Dies ist, wie in Kapitel 4.1 beschrieben, besonders dann notwendig, wenn die Applikation, in der die Relationen erfasst wurden, keine vollständige Nachverfolgung aller Aktionen und Prozesse, die zur Entstehung einer Beziehung führen können, erlaubt.

Abhängig vom Relationstyp und vom Typ der Dokumente gibt es verschiedene mögliche Ansätze. In den beiden betrachteten Fällen ist das Validierungsproblem auf die Erkennung von Wiederverwendung abbildbar. Zur Validierung von textbasierten Elementrelationen bieten sich hier konkret Algorithmen aus dem Bereich der Ähnlichkeitssuche ("Fuzzy-String-Matching") an. In diesem Kapitel werden bestehende Algorithmen erläutert und eine Erweiterung für einen dieser Algorithmen vorgeschlagen (Abschnitt 5.3) und evaluiert (Abschnitt 5.4).

Objektbasierte Elementrelationen, die zum Beispiel entstehen, wenn eine Folie oder ein Bild zwischen zwei Dokumenten kopiert wird, müssen nicht inhaltsbasiert validiert werden. Eine Validierung kann hier über während der Erfassung der Relation vergebene und persistierte IDs erfolgen. Selbiges gilt für Aggregationsrelationen. Bei Verknüpfungsrelationen ist die Identifizierbarkeit durch die Verknüpfung und deren identifizierenden Charakter gegeben.

Die Validierung von Variantenrelationen kann mit Hilfe von Fingerprinting-Algorithmen erfolgen und wird in Kapitel 6 erläutert.

5.1 Grundlagen

Im Folgenden werden zunächst grundlegende Begriffe erläutert und in dem gegebenen Zusammenhang relevante Metriken beziehungsweise Ähnlichkeits- und Distanzmaße sowie sinnvolle Vorverarbeitungsschritte beschrieben. Diese sind für die in diesem Kapitel beschriebenen String-Matching-Ansätze und für die in Kapitel 6 beschriebenen Fingerprinting-Ansätze weitestgehend identisch. Deshalb werden sie in diesem Kapitel zusammenfassend erläutert.

5.1.1 Grundlegende Begriffe

Grundlegende, hier benötigte Begriffe für die Betrachtung textbasierter Dokumente sind "Token" und "Shingle".

Tokens sind das Ergebnis der Unterteilung einer Zeichenkette in ihre Einzelteile. Am häufigsten werden Tokens auf Wortebene oder auf Buchstabenebene verwendet - ein Token ist dabei ein einzelnes Wort beziehungsweise ein einzelner Buchstabe - es sind jedoch auch andere Arten der Unterteilung denkbar. Natürlichsprachliche Texte werden zumeist in Tokens auf Wortebene zerlegt, während bei anderen Arten von Zeichenketten, wie zum Beispiel DNA-Sequenzen, eine Zerlegung in Tokens auf Buchstabenebene

sinnvoller ist. Im Folgenden sind, wenn die Tokens eines Textes oder Dokuments erwähnt werden, die einzelnen Worte des Textes gemeint.

Ein **Shingle** ist ebenfalls das Ergebnis einer Zerlegungsoperation eines Textes. Ein Shingle ist der Inhalt eines Fensters der Größe n, das schrittweise über einen Text geschoben wird. Hierbei ist n die Anzahl der Tokens pro Fenster. Alle Shingles eines Textes werden extrahiert, indem der Inhalt des Fensters an jeder möglichen Position im Text extrahiert wird. Ein Shingle enthält also jeweils n im Text aufeinanderfolgende Tokens. Zwei im Text aufeinanderfolgende Shingles überlappen sich und stimmen in $n - 1$ Tokens überein.

5.1.2 Vorverarbeitung von Zeichenketten

Um die Effizienz der später vorgestellten String-Matching-Algorithmen und Fingerprinting-Techniken zu erhöhen, können verschiedene Vorverarbeitungsschritte auf den gegebenen Zeichenketten durchgeführt werden. Im Folgenden werden die Techniken beschrieben, die in dieser Arbeit zum Einsatz kommen.

- **Säubern:** Hierbei werden ungewollte Zeichen, zum Beispiel Markup, oder Symbole aus dem Text entfernt. Zumeist werden hierbei auch Satzzeichen aus dem Text entfernt.
- **Groß-Kleinschreibung:** Durch diesen Prozessschritt wird die Groß- bzw. Kleinschreibung aller Wörter angeglichen. Meistens wird die Kleinschreibung gewählt.
- **Kanonikalisierung:** Hierbei werden verschiedene Ausdrücke in eine einheitliche Darstellung gebracht. Dies ist beispielsweise bei der Darstellung von Dezimalzahlen, Datumsangaben oder Preisangaben notwendig. Es wird ein einheitliches (kanonisches) Format verwendet.
- **Stoppwortfilterung:** Mit Hilfe von Stoppwortlisten können Worte aus dem Text entfernt werden, deren Informationsgehalt - zum Beispiel aufgrund der großen Häufigkeit ihres Auftretens - gering ist. Dies gilt beispielsweise für die meisten Konjunktionen, Artikel, Präpositionen oder Pronomen.
- **Stammbildung / Lemmatisierung:** Hierbei wird ein Verb oder Substantiv auf seinen Wortstamm zurückgebildet. Dies erleichtert das Matching, da unterschiedliche Formen eines Substantivs oder Verbs nicht mehr berücksichtigt werden müssen. Allerdings führt es auch zu einer erhöhten Unschärfe, da vormals unterschiedliche Formen des Wortes nun als gleich angesehen werden. Hier ist abzuwägen, ob es in dem gegebenen Anwendungsfall auf die Form oder die Bedeutung eines Wortes ankommt.
- **Zerlegung:** Hierbei wird die Zeichenkette in Tokens zerlegt.

Je nach Anwendungsfall ist es sinnvoll, eine Kombination aus verschiedenen Vorverarbeitungstechniken zu verwenden.

5.1.3 Ähnlichkeits- und Distanzmaße

Eine Wiederverwendungsbeziehung, wie sie in 2.4.3 definiert ist, impliziert grundsätzlich eine Verwandtschaft zwischen den beiden in Beziehung stehenden Dokumenten. Diese Verwandtschaft lässt sich mit

Hilfe eines Ähnlichkeits- bzw. Distanzmaßes ausdrücken. Im Fall einer Variantenrelation besteht diese Ähnlichkeit zwischen den Dokumenten selbst, bei Elementrelationen zwischen den jeweiligen Artefakten und bei Aggregationsrelationen zwischen einem Artefakt und einem Dokument beziehungsweise Medienobjekt. In der Literatur gibt es verschiedene Maße, um die Ähnlichkeit oder Distanz zweier Objekte zu modellieren. Die für diese Arbeit relevanten Maße werden im Folgenden erläutert.

Jaccard-Maß

Das Jaccard-Maß (oder Jaccard-Koeffizient) ist ein Maß für die Ähnlichkeit zweier Mengen [Jac01]. Es ist definiert als:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} \tag{5.1}$$

Die Schnittmenge beider Mengen, also die Elemente, die beide Mengen gemeinsam haben, wird durch die Vereinigungsmenge - also die Menge aller Elemente aus A oder B - dividiert. Dadurch ergeben sich Werte für die Ähnlichkeit der beiden Mengen im Bereich $[0,1]$. Dieses Maß normiert die Ähnlichkeit auf die Summe der Elemente in beiden Mengen. Es spielt also keine Rolle, ob A und B eine unterschiedliche Anzahl an Elementen haben und wie groß dieser Unterschied ist. Somit ist das Maß symmetrisch, es gilt:

$$J(A,B) = J(B,A) \tag{5.2}$$

Laut der Definition einer Menge ist jedes Element einer Menge einzigartig. Dies führt dazu, dass Duplikate ignoriert werden. Es kann jedoch auch sinnvoll sein, Duplikate von Elementen für die Ähnlichkeitsberechnung zu berücksichtigen. Hierzu wird das Jaccard-Maß auf *Multimengen* angewendet. Eine Multimenge ist eine Menge, in der ein Element mehrfach vorkommen kann. Sie ist definiert als ein Paar (M,f), wobei M eine Menge und f eine Abbildung von $M \rightarrow \mathbb{N}$ ist. Die Abbildung f gibt an, wie oft jedes Element von M in der Multimenge auftritt [Syr01]. Um das Jaccard-Maß für zwei Multimengen zu berechnen, müssen die Schnittmenge, Vereinigungsmenge und deren Beträge berechnet werden können. Für die Kardinalität einer Multimenge $\mathbb{A} = (A,f)$ ergibt sich dabei:

$$|\mathbb{A}| = \sum_{a \in A} f(a) \tag{5.3}$$

Die Schnittmenge $\mathbb{C} = (A,h)$ von zwei Multimengen $\mathbb{A} = (A,f)$ und $\mathbb{B} = (A,g)$ ist definiert als:

$$h(a) = min(f(a), g(a)) \quad \forall a \in A \tag{5.4}$$

Für die Vereinigungsmenge $\mathbb{D} = (A, k)$ der beiden Multimengen gilt:

$$k(a) = max(f(a), g(a)) \quad \forall a \in A \tag{5.5}$$

Um die Kardinalität einer Multimenge zu berechnen, werden also die Häufigkeiten aller Elemente der Multimenge aufaddiert. Die Schnittmenge wird gebildet, indem für jedes Element der beiden Mengen die minimale Häufigkeit gewählt wird. Bei der Vereinigungsmenge wird dementsprechen der Maximalwert gewählt. Die Schnittmenge der zwei Multimengen $\{1, 1, 1, 2, 2, 3, 5, 8\}$ und $\{1, 1, 2, 5, 9\}$ ist also $\{1, 1, 2, 5\}$, während sich für die Vereinigungsmenge $\{1, 1, 1, 2, 2, 3, 5, 8, 9\}$ ergibt. Wenn man das Jaccard-Maß auf Multimengen überträgt, ergibt sich folgendes Ähnlichkeitsmaß:

$$J(\mathbb{A}, \mathbb{B}) = \frac{|\mathbb{A} \cap \mathbb{B}|}{|\mathbb{A} \cup \mathbb{B}|} \tag{5.6}$$

Cosinus-Maß

Ein weiteres Maß, welches für die Bestimmung der Ähnlichkeit von Dokumenten genutzt werden kann, ist das häufig im Vektorraum-Retrieval genutzte *Cosinus-Maß* (z.B. [Fer03]). Es definiert die Ähnlichkeit zweier Vektoren als Cosinus des Winkels zwischen diesen Vektoren:

$$Sim(A, B) = \cos(\theta) = \frac{A \cdot B}{||A|| \cdot ||B||} \tag{5.7}$$

Das Cosinus-Maß nimmt Werte im Bereich $[-1, 1]$ an, wobei ein Wert von -1 ausdrückt, dass der Vektor in die entgegengesetzte Richtung zeigt. Bei einem Wert von null sind die Vektoren unabhängig. Wiederum gilt:

$$Sim(A, B) = Sim(B, A) \tag{5.8}$$

Es handelt sich also ebenfalls um ein symmetrisches Ähnlichkeitsmaß.

Abdeckungsmaß

Beide zuvor vorgestellten Ähnlichkeitsmaße sind symmetrisch. Bei der Wiederverwendung von Artefakten ist es jedoch sehr häufig der Fall, dass Quell- und Zieldokument unterschiedliche Länge aufweisen. Ein symmetrisches Ähnlichkeitsmaß reflektiert einen solchen Längenunterschied nicht. Wenn zum Beispiel ein kurzes Dokument nahezu vollständig in einem längeren Dokument wiederverwendet wird, kann ein solches Maß nur unzureichende Aussagen über die Ähnlichkeit treffen. Es wird ein Maß benö-

tigt, welches Unterschiede in der Länge der Dokumente berücksichtigt. Deshalb wird ein asymmetrisches Ähnlichkeitsmaß wie folgt definiert [Bro97]:

$$S(A,B) = \frac{|A \cap B|}{|A|} \tag{5.9}$$

Die *Abdeckung* der Menge A durch die Menge B ist also der Quotient aus der Schnittmenge von A und B und der Anzahl der Elemente der Menge A. Analog gilt für die Abdeckung von B durch A:

$$S(B,A) = \frac{|A \cap B|}{|B|} \tag{5.10}$$

Das Maß gibt wieder, bis zu welchem Grad ein Dokument in einem anderen enthalten ist. Ein Unterschied in der Länge der Dokumente wird demnach von diesem Maß berücksichtigt. Das Abdeckungsmaß kann auch auf Multimengen angewendet werden, um mehrfach auftretende Elemente für die Berechnung der Ähnlichkeit zu berücksichtigen.

Distanzmaße

Neben den Ähnlichkeitsmaßen gibt es auch *Distanzmaße*. Der durch ein Distanzmaß angegebene Wert wächst mit sinkender Ähnlichkeit der verglichenen Objekte. Ein oft für den Vergleich von Binärdaten, aber auch Zeichenketten eingesetztes Maß ist die sogenannte *Hamming-Distanz* [Ham50]. Die Hamming-Distanz gibt an, in wievielen Stellen sich zwei Zeichenketten unterscheiden. Sie ist für zwei Zeichenketten x und y eines endlichen Alphabets definiert als:

$$\Delta(x,y) := \sum_{x_i \neq y_i} 1, \quad i = 1,...,n \tag{5.11}$$

Bei binären Daten kann die Berechnung der Hamming-Distanz auch durch Durchführung einer XOR Operation auf den beiden Bitstrings und nachfolgendes Zählen der im Resultat vorhandenen Einsen erfolgen.

Die *Levenshtein-Distanz* [Lev66] beruht auf einem ähnlichen Prinzip wie die Hamming-Distanz. Sie bemisst den Unterschied zweier Zeichenketten und drückt aus, wie viele Operationen der Typen "Einfügen", "Löschen" oder "Ersetzen" auf Buchstabenebene mindestens notwendig sind, um die eine Zeichenkette in die andere zu überführen. Um zum Beispiel das Wort "Buche" in das Wort "suchen" zu überführen, sind folgende zwei Operationen notwendig:

1. suche (Ersetze 'B' durch 's')
2. suchen (Füge 'n' ein)

Die Levenshtein-Distanz bietet demnach ein Maß, um auszudrücken, wie weit zwei Zeichenketten auf lexikalischer Ebene voneinander entfernt sind.

5.2 Existierende String-Matching-Algorithmen

String-Matching-Algorithmen beschäftigen sich mit dem Problem, eine gegebene Zeichenkette in einem bestimmten Text beziehungsweise einer Menge von Texten zu finden. Im Information Retrieval können sie beispielsweise dazu genutzt werden, um zu beurteilen, ob eingegebene Suchbegriffe in einem Dokument vorkommen. In dieser Arbeit werden sie dazu genutzt, Elementrelationen, die durch die Wiederverwendung von Text zustande gekommen sind, zu validieren. Zunächst werden existierende Algorithmen zur Lösung des Problems beschrieben und auf die Anwendbarkeit in diesem Szenario hin beurteilt, bevor der in dieser Arbeit verwendete ShingleCloud-Ansatz beschrieben wird. Im Folgenden werden drei bekannte und häufig verwendete Verfahren zum Matching von Strings beschrieben: Der N-Gram-Overlap-Algorithmus ("n-Gramm-Überlappung") vorgestellt von Caroline Lyon in [LMD01], Greedy-String-Tiling (GST) beschrieben von Michael Wise in [Wis93] und der Longest-Common-Subsequence-Algorithmus als klassisches Informatikproblem [Mai78].

5.2.1 N-Gram-Overlap

Bei diesem Ansatz werden die zu vergleichenden Texte jeweils in Shingles der Länge n zerlegt (auch n-Gramme genannt). Für jeden Text ergibt sich damit eine Multimenge von n-Grammen. Zum Beispiel ergibt sich aus dem Text "Da steh' ich nun, ich armer Tor" folgende Menge von 3-Grammen:

[Da steh' ich] | [steh' ich nun] | [ich nun ich] | [nun ich armer] | [ich armer Tor].

Im Original nach Lyon wird der Algorithmus als mengentheoretischer Ansatz durchgeführt. Das hat zur Folge, dass die beiden n-Gramm-Multimengen der zu vergleichenden Dokumente als Mengen betrachtet werden, in denen jedes n-Gramm nur einmal vorhanden ist. Die Ähnlichkeit der beiden Texte wird dann mit Hilfe des Jaccard-Maßes berechnet: Die Anzahl der n-Gramme, welche in beiden Texten vorkommen, geteilt durch die Anzahl aller vorkommenden n-Gramme. Als guter Wert zur Erkennung von Übereinstimmungen hat sich Fenstergröße drei erwiesen [LMD01, BCR09, SC08].

Es gibt verschiedene Möglichkeiten, dieses Verfahren zu variieren. Dazu zählt die Betrachtung der n-Gramm-Mengen der Ausgangstexte als Multimengen, das heißt also die Berücksichtigung von n-Gramm-Duplikaten. Des weiteren wird häufig anstatt des symmetrischen Jaccard-Maßes das asymmetrische Abdeckungsmaß (siehe 5.1.3) verwendet. Weiterhin kann auch das Cosinus-Maß zur Berechnung der Ähnlichkeit herangezogen werden, wenn die resultierenden Shingles als Vektor betrachtet werden. Der beschriebene Algorithmus erzeugt für ein Dokument D mit $|D|$ Tokens die folgende Anzahl an n-Grammen:

$$M(D) = |D| - n + 1 \tag{5.12}$$

Beim Multimengenansatz entspricht dies auch der Menge der tatsächlich zu speichernden n-Gramme beziehungsweise Hashwerte. Bei einem mengenbasierten Ansatz ist die Gesamtzahl der n-Gramme entsprechend der Anzahl der mehrfach auftretenden n-Gramme geringer. Dieser Ansatz eignet sich gut für eine Anwendung auf großen Mengen von Dokumenten. Um die Laufzeit zu verbessern, wird hierbei ein

inverser Index (siehe [BYRN99]) über die extrahierten n-Gramme erzeugt. Der Ansatz selbst besitzt eine Komplexität von $O(n)$ [LBPS08].

5.2.2 Longest-Common-Subsequence

Das diesem Algorithmus zu Grunde liegende Problem ist ein klassisches Problem in der Informatik. Es wird die längste gemeinsame Sequenz (LCS) zweier gegebener Strings gesucht. Im Gegensatz zum längsten gemeinsamen *Teilstring* können hier ungleiche Stellen übersprungen werden. Bei den gegebenen Strings "ABAEG" und "ABUZAG" ist der längste gemeinsame Teilstring "AB". Die längste gemeinsame Teilsequenz ist jedoch "ABAG". Wenn Quell- und Zieltext gegeben sind, besitzen Ansätze zur Bestimmung der LCS typischerweise die Komplexität $O(n^2)$ [LBPS08]. Die Ähnlichkeit ergibt sich aus dem Quotienten der zweifachen Länge der längsten gemeinsamen Teilsequenz und der Gesamtlänge der beiden Strings:

$$S(A,b) = \frac{2 * |LCS(A,B)|}{|A| + |B|} \tag{5.13}$$

Auch hier kann die Ähnlichkeit wiederum asymmetrisch berechnet werden, indem die einfache Länge des LCS statt durch die Gesamtlänge beider Strings nur durch die Länge von jeweils einem der Texte dividiert wird.

5.2.3 Greedy-String-Tiling

Beim Greedy-String-Tiling (GST) [Wis93] werden aus Quell- und Zieltext sogenannte Tiles ("Fliese/Ziegel") gebildet. Dies sind jeweils Teilstrings, die in beiden gegebenen Texten vorkommen. Unter Angabe einer Mindestlänge wird ein sogenanntes Tiling zweier Texte erzeugt. Das bedeutet, Quell- und Zieltext werden in gleiche Substrings der gegebenen Mindestlänge unterteilt. Die Greedy-String-Tiling-Heuristik versucht nun das maximale Tiling, also das Tiling mit der größten Abdeckung, zwischen zwei Texten zu finden. Das Problem ist NP schwer, die Worst-Case-Komplexität der Heuristik beträgt $O(n^3)$ [LBPS08]. Die Heursitik sucht zunächst nach Übereinstimmungen maximaler Länge und markiert die entsprechenden Segmente in den gegebenen Texten. Für die weitere Suche werden bereits markierte Segmente ignoriert. Der Algorithmus wird solange durchgeführt, bis keine Tiles mit der gegebenen Minimallänge mehr gefunden werden. GST erreicht die maximale Abdeckung der beiden Strings bei einer Minimalgröße von eins für die Tiles. Es ist jedoch nicht immer sinnvoll, die Minimallänge so gering zu wählen, da dadurch GST in einen einfachen Wortvergleichsalgorithmus überführt wird. Der Algorithmus zieht zudem größere Tiles einer optimalen Abdeckung vor. Eine große Übereinstimmung, die zum Beispiel acht Terme beinhaltet, wird zwei kleineren mit fünf beziehungsweise vier Tokens vorgezogen, obwohl letzteres Tiling eine bessere Abdeckung gewährleistet.

Schließlich kann zur Berechnung der Ähnlichkeit die Gesamtlänge der gefundenen Tiles durch die Länge des jeweiligen Dokumentes geteilt werden. Für die Ähnlichkeit zweier Texte A und B ergibt sich also:

$$S_{gst}(A,B) = \frac{\sum_{i \in Tiles} L_i}{|A|} \tag{5.14}$$

Hierbei ist L_i die Länge des Tiles i in der durch das Tiling entstandenen Menge von Tiles. Wiederum handelt es sich um ein asymmetrisches Maß, welches dementsprechend auch im Verhältnis zur Länge von Dokument B berechnet werden kann.

5.2.4 Beurteilung

Alle beschriebenen Ansätze wurden in verschiedenen Anwendungsszenarien getestet und evaluiert (z.B. [CGPW02, LBPS08, PMP02]). Die Qualität eines Ansatzes ist hierbei immer von dem Szenario selbst und den jeweiligen Auflagen abhängig. Soll zum Beispiel ein Suchstring für das Retrieval von Dokumenten aus einer sehr großen Dokumentenmenge verglichen werden, so ist die Performanz des Ansatzes wichtig, und es sollte ein Ansatz mit geringer Komplexität gewählt werden. Ist die Performanz zweitrangig, kann auch ein anderer Algorithmus, der qualitativ bessere Ergebnisse produziert, ausgewählt werden.

In dem hier betrachteten Anwendungsfall soll ein wiederverwendeter Textabschnitt mit einem Dokument abgeglichen werden, um beurteilen zu können, ob dieser noch in dem Dokument enthalten ist. Ein solcher Text wird im Zieldokument häufig Änderungen unterzogen. Teile werden gelöscht, neue hinzu- oder eingefügt. Außerdem wird ein wiederverwendeter Abschnitt häufig dem Fluss des Textes, in den er eingefügt wurde, angepasst, um ihn lesbarer zu machen und Stilbrüche oder Mosaikeffekte [IRC03] zu vermeiden. Der gesuchte Algorithmus muss folglich mit dem Löschen, Einfügen und Verändern der Reihenfolge der entsprechenden Textabschnitte umgehen können. Zudem muss eine Grenze bestimmt werden, ab welchem Grad an Änderung oder Löschung von Text eine Beziehung noch als valide angesehen werden kann. Da letzteres von Nutzer zu Nutzer verschieden sein kann, muss der gewählte Algorithmus diesem mit entsprechender Parametrisierbarkeit und Anpassbarkeit Rechnung tragen. Im Folgenden werden die vorgestellten Algorithmen auf ihre Anwendbarkeit in dem gegebenen Szenario hin beurteilt.

Der **Longest-Common-Subsequence**-Algorithmus ist nicht gut geeignet für diesen Anwendungsfall, da er die Reihenfolge der gegebenen Strings erhält, also demnach anfällig für Veränderungen der Reihenfolge der Texte ist. Das ist nicht nur für Veränderungen der Reihenfolge auf Wortebene der Fall, sondern auch, wenn die Reihenfolge ganzer Sätze oder Nebensätze verändert wird. Wenn beispielsweise die Position eines Nebensatzes im Vergleich zum Quelldokument verändert wird, ergibt sich nach der LCS-Methode eine viel zu geringe Ähnlichkeit.

Greedy-String-Tiling kann für den gegebenen Anwendungsfall in begrenztem Umfang genutzt werden, ist jedoch relativ komplex. GST liefert erfahrungsgemäß gute Ergebnisse. Da es jedoch vorkommen kann, dass ein Dokument sehr viele texthaltige Elementrelationen besitzt und eine Validierung dieser auch während der Bearbeitung des Dokuments durch den Nutzer möglich sein soll, ist ein Einsatz von GST in dem gegebenen Anwendungsfall nicht sinnvoll.

Navarro beschreibt in [Nav01] eine Vielzahl von auf der **Levenshtein-Distanz** basierender Verfahren, die unterschiedliche Laufzeit und unterschiedlichen Speicherbedarf besitzen und vergleicht diese. Diese Verfahren eignen sich jedoch vorrangig für die Fuzzy-Suche kurzer Textsequenzen, bei der Strings trotz Tippfehlern oder unterschiedlicher Schreibweise als übereinstimmend erkannt werden, oder für die Rechtschreibkorrektur - die Wörter mit der geringsten Levenshtein-Distanz zum eingegebenen (falschen) Wort werden vorgeschlagen. Sie werden für den hier betrachteten Anwendungsfall nicht berücksichtigt.

N-Gram-Overlap hat die geringste Komplexität, liefert aber dennoch robuste Ergebnisse. Eine einfache Berechnung der Abdeckung oder Ähnlichkeit reicht jedoch nicht aus. Neben der einfachen Aussage, ob eine Relation valide ist, soll das Ergebnis insbesondere durch den Nutzer nachvollziehbar sein. Dies kann durch eine Visualisierung der gefundenen Überschneidungen ermöglicht werden. Dadurch ist eine Feedback-Schleife denkbar, die anhand der Rückmeldung des Nutzers zu der Beurteilung einer Beziehung die Parameter des Algorithmus anpasst. Die Abdeckung der beiden Strings muss in einer Form modelliert werden, die sowohl menschen- als auch maschineninterpretierbar dargestellt werden kann. Aufbauend auf N-Gram-Overlap wurde ein Algorithmus zur Validierung textbasierter Elementrelationen entwickelt, der im Folgenden vorgestellt wird.

5.3 ShingleCloud

ShingleCloud wurde für die Validierung von Elementrelationen im LIS.KOM-Framework entwickelt (siehe [ML10] für Details) und von Mittelbach et al. im Rahmen des Holinshed-Projektes angepasst und weiterentwickelt [CM09a, CM09b] (siehe Abschnitt 5.5.3). Das Ziel des Designs von ShingleCloud ist ein performanter und flexibler Algorithmus, der erkennt, ob sich ein Textabschnitt beliebiger Länge in gegebenenfalls angepasster Form in einem gegebenen Dokument befindet. Der Algorithmus ist eine Erweiterung des N-Gram-Overlap Ansatzes. Der wiederverwendete Text, der gefunden werden soll, wird im Folgenden als "Nadel" (N) bezeichnet, während das Dokument, in dem gesucht wird, "Heuhaufen" (H) genannt wird. Es gilt $|N| \leq |H|$. Die ersten Schritte des Algorithmus entsprechen dem Vorgehen beim N-Gram-Overlap-Algorithmus. Nadel und Heuhaufen werden verschiedenen Vorverarbeitungsschritten unterzogen. Dazu zählen Säuberung, Kanonikalisierung sowie optional Stoppwortfilterung. Daraufhin werden beide Texte in n-Gramme zerlegt und geordnet gespeichert, zum Beispiel als Listen. Das Zwischenergebnis des Algorithmus ist ein String, bestehend aus Nullen und Einsen (die "Shingle-Cloud"), der wie folgt erzeugt wird:

1. Die n-Gramme des Heuhaufens werden entsprechend ihrer Reihenfolge durchlaufen.

2. Wenn das jeweilige n-Gramm des Heuhaufens in der Liste der n-Gramme der Nadel enthalten ist, wird eine "1" an die Shingle-Cloud angefügt, ansonsten eine "0".

Dadurch entsteht eine Abbildung des Auftretens der Nadel im Heuhaufen, welche durch ein einmaliges Durchlaufen der n-Gramme des Heuhaufens erzeugt werden kann. Die entstandene Shingle-Cloud kann nun interpretiert werden, um zu entscheiden, ob die Nadel als im Heuhaufen enthalten gewertet wird.

Im Folgenden werden zunächst einige grundlegende Betrachtungen zur Bedeutung unterschiedlicher Muster in resultierenden Shingle-Clouds dargelegt, bevor mögliche Interpretationen anhand einer Reihe von Beispielen erläutert werden. Die Bedeutung einer einzelnen "1" beziehungsweise "0" in der resultie-

renden Shingle-Cloud ist von zwei Faktoren abhängig, nämlich von der gewählten n-Gramm Länge und den in der Shingle-Cloud vorhergehenden Werten. So bedeutet eine "1" eine Übereinstimmung von "n" Tokens aus Nadel und Heuhaufen, wenn der vorhergehende Wert in der Shingle-Cloud keine "1" ist. Ist dies jedoch der Fall, so bedeutet die "1", dass lediglich ein weiterer Token übereinstimmt. Bei langen Sequenzen von Einsen oder Nullen in der Shingle-Cloud fällt die Fenstergröße nicht mehr ins Gewicht. Eine Sequenz von k Einsen bedeutet, dass $k+n$ Tokens aus Nadel und Heuhaufen übereinstimmen. Sie wird im folgenden als *Übereinstimmung* (Match) bezeichnet. Die Bedeutung einer Null lässt sich auf ähnliche Weise beschreiben. Wenn eine Null auf eine Eins folgt, so bedeutet dies, dass ein Token aus dem Heuhaufen nicht in der Nadel enthalten ist. Für jede weitere Null kommt ein weiterer nicht übereinstimmender Token hinzu. Einzige Ausnahme bildet hier das erste Zeichen der Shingle-Cloud. Wenn es sich dabei um eine Null handelt, so bedeutet diese, dass "n" Tokens nicht übereinstimmen. Im folgenden werden verschiedene Muster, die eine Shingle-Cloud annehmen kann, analysiert. Dabei wird eine längere Sequenz von Nullen in der Form 0^u notiert.

$$0^u 111111111111111110^v$$

Dieses Muster bedeutet, dass eine längere Sequenz der Nadel im Heuhaufen enthalten ist. Ob es sich dabei um die gesamte Nadel handelt, hängt von der Länge der Nadel und der Anzahl der Einsen in der Shingle-Cloud ab. Ein solches Muster kommt zustande, wenn die Nadel nach dem Einfügen in den Heuhaufen nicht verändert wird, aber auch wenn Teile vom Anfang oder Ende der Nadel entfernt werden. Die Nadel ist dann ganz im Heuhaufen enthalten, wenn gilt: $|N| = k+n-1$.

$$0^u 11111111110^v 11111111111110^w$$

Ein solches Muster entsteht, wenn innerhalb der Nadel etwas in den Heuhaufen eingefügt wurde. Dabei kann es sich um einen Nebensatz oder auch einen ganzen Abschnitt handeln. Je größer v desto länger der innerhalb der Nadel eingefügte Text. Alternativ kann ein solches Muster auch entstehen, wenn die Nadel an zwei verschiedenen Stellen in denselben Heuhaufen eingefügt wird. Dies kann meist anhand der Länge der Übereinstimmungen im Vergleich zur Länge der Nadel unterschieden werden.

$$0^u 1111111000111111001111100011111110^v$$

Dieses Muster entsteht wenn, innerhalb der Nadel einzelne Token verändert, gelöscht oder eingefügt werden, die Nadel im Großen und Ganzen aber erhalten bleibt. Eine Sequenz von drei Nullen ergibt sich bei einer Shingle-Größe von drei durch das Einfügen eines neuen Tokens in die Nadel. Durch Löschen eines Tokens ergibt sich eine Sequenz von zwei Nullen. Zwei Nullen ergeben sich auch, wenn man einen bereits in der Nadel bestehenden Token direkt dahinter erneut einfügt. Dieser Fall kann jedoch in der Regel ausgeschlossen werden. Bei einer Shingle-Größe von drei bedeutet das gezeigte Muster, dass nach

neun Tokens der Nadel ein zusätzlicher Term eingefügt wurde, dann nach wiederum acht weiteren Tokens einer aus dem original Wortlaut der Nadel entfernt und nach sieben weiteren Tokens schließlich wieder ein neuer Token eingefügt wurde. Mit Hilfe dieser Informationen ist es möglich, die Übereinstimmungen zwischen Nadel und Heuhaufen sehr genau zu bestimmen und auch zu visualisieren. Eine automatisierte Entscheidung über die Validität einer gegebenen Relation kann dann über die Existenz einer Übereinstimmung getroffen werden. Ist eine genügend lange Übereinstimmung der Nadel mit dem Heuhaufen vorhanden, wird die Relation als gültig eingestuft. Ist dies nicht der Fall, wird sie als ungültig gewertet.

Alternativ kann als Maß für die Bestimmung der Gültigkeit auch die Abdeckung der Nadel durch den Heuhaufen genutzt werden. Diese ist wie folgt für eine Nadel N und einen Heuhaufen H und die Shingle-Größe n definiert:

$$S(N,H) = \frac{f * n + (k - f)}{|N|} \tag{5.15}$$

Hierbei ist f die Anzahl der Einsen, deren Vorgänger in der Shingle-Cloud keine Eins ist und k die Gesamtzahl der Einsen in der Shingle-Cloud. Wenn ein genügend großer Anteil der Nadel im Heuhaufen enthalten ist, kann eine Relation als gültig bewertet werden.

Es gibt drei verschiedene Parameter, mit deren Hilfe der ShingleCloud Algorithmus angepasst werden kann:

Die **Shingle-Größe** n: Durch die Fenstergröße, mit der die Shingles erzeugt werden, wird bestimmt, ab welcher Länge eine Übereinstimmung auf Tokenbasis eine Eins in der Shingle-Cloud ergibt. Bei sehr großen n ist der Algorithmus sehr strikt, und bereits kleine Änderungen bewirken ein Auftreten von vielen Nullen in der Shingle-Cloud. Wenn die Fenstergröße sehr klein ist, zum Beispiel $n = 1$, dann häufen sich zufällige Übereinstimmungen und verfälschen das Ergebnis. Eine typische Fenstergröße ist, wie auch bei dem zu Grunde liegenden N-Gram-Overlap, $n = 3$.

Die **Minimalzahl von Einsen in einer Übereinstimmung**: Dieser Parameter gibt an, ab welcher Anzahl von Einsen eine Übereinstimmung (d.h. eine Sequenz von Einsen in der Shingle-Cloud) gewertet wird. Dadurch kann die Zahl der zufälligen Treffer verringert werden. Wird die Minimalzahl der Einsen auf Null gesetzt, wird jede einzelne Eins in der Shingle-Cloud als Übereinstimmung gewertet. Je isolierter eine Eins in der Shingle-Cloud ist, desto größer ist jedoch die Wahrscheinlichkeit, dass es sich dabei um eine zufällige Übereinstimmung handelt. Mit Hilfe dieses Parameters kann die Zahl der zufälligen Übereinstimmungen verringert werden, auch wenn die Shingle-Größe klein ist.

Die **Maximalzahl von Nullen innerhalb einer Übereinstimmung**: Mit Hilfe dieses Parameters kann die Anfälligkeit des Algorithmus für das Löschen und Einfügen von Text verändert werden. Wenn dieser Parameter auf den Wert n gesetzt wird, hat das zur Folge, dass der Effekt des Einfügens eines einzelnen Tokens egalisiert wird. Wenn also eine Anzahl Nullen, die kleiner als die gegebene Maximalzahl ist, in der Shingle-Cloud von zwei Einer-Sequenzen eingeschlossen wird, werden die beiden Einer-Sequenzen als eine Übereinstimmung interpretiert.

Um aus einer Shingle-Cloud die Ähnlichkeit zweier beliebiger Texte zu berechnen, wird der eine Text als Nadel und der andere als Heuhaufen behandelt. Die Shingle-Cloud wird wie beschrieben ermittelt.

Dann wird für jede Übereinstimmung wie beschrieben die Anzahl der Tokens berechnet. Schließlich wird die Summe der übereinstimmenden Tokens aller Matches durch die Gesamtzahl der Tokens der Nadel beziehungsweise des Heuhaufens geteilt, um jeweils die Abdeckung zu ermitteln, wie in Gleichung 5.15 gegeben. Da eine Nadel auch mehrfach in einem Heuhaufen auftreten kann, können hierbei auch Ähnlichkeitswerte $S(N,H) > 1$ auftreten. Ist dies nicht gewünscht, können übereinstimmende Shingles jeweils aus der Nadel entfernt werden.

5.4 Evaluationsszenario und Methodologie

Im Folgenden wird die Evaluation des für die Validierung von Elementrelationen entwickelten Shingle-Cloud-Algorithmus beschrieben. Hierfür werden zunächst die Methodologie sowie die verwendeten Korpora beschrieben. Die angewandte Methodologie sowie die verwendeten Korpora stimmen auch hier mit der Evaluation des in Kapitel 6 dargestellten MiLe-Fingerprinting-Ansatzes überein, so dass sie hier zusammenfassend dargestellt werden.

5.4.1 Definitionen und Vorgehen

Einige der durchzuführenden Evaluationen lassen sich auf Kategorisierungsprobleme zurückführen. Ein solches Problem gibt Objekte und Kategorien vor. Der zu evaluierende Ansatz muss dann einem Objekt jeweils eine der Kategorien zuordnen. Bei Kategorisierungsproblemen, die zur Evaluation von Algorithmen verwendet werden, sind die korrekten Kategorien der Objekte bekannt und die Qualität des Ansatzes wird anhand der von ihm vorgenommenen Einordnung im Vergleich mit der gegebenen Einordnung ermittelt. Hierfür werden die für Kategorisierungsprobleme üblichen Maße zur Bestimmung der Qualität eines Algorithmus angewendet (siehe [MKSW99] und [BYRN99]). Dazu zählen *Trefferquote* (Recall), *Genauigkeit* (Precision) und *Ausfallquote* (Fallout) sowie das F1-Maß als harmonisches Mittel aus Trefferquote und Genauigkeit.

Die **Trefferquote** (Recall) eines Verfahrens gibt an, welchen Anteil der Objekte einer Kategorie das Verfahren richtig einordnet. Wenn R die Menge der relevanten Dokumente einer Kategorie darstellt und P die Menge der durch einen Ansatz in diese Kategorie eingeordneten Objekte, so ergibt sich für die Trefferquote des Verfahrens:

$$Recall = \frac{|R \cap P|}{|R|} \tag{5.16}$$

Die **Genauigkeit** (Precision) gibt an, welcher Anteil der in eine Kategorie eingeordneten Objekte korrekt klassifiziert sind. Also ist sie definiert als:

$$Precision = \frac{|R \cap P|}{|P|} \tag{5.17}$$

Die **Ausfallrate** (Fallout) schließlich gibt an, welchen Anteil der insgesamt für eine Kategorie unpassenden Objekte dieser Kategorie zugeordnet wurden. Sie ist für die Bestimmung der Qualität eines Ka-

tegorisierungsalgorithmus nicht zwingend notwendig. Im Gegensatz dazu genügt es jedoch nicht, nur Genauigkeit oder nur die Trefferquote zu betrachten, um die Qualität eines Ansatzes zu bestimmen. So hat zum Beispiel ein Ansatz, der einer Kategorie einfach alle vorhandenen Objekte zuweist, zwar eine sehr große Trefferquote, aber eine sehr geringe Genauigkeit. Um dennoch nicht immer beide Maße angeben zu müssen, kann auch das **F1-Maß** als harmonischer Mittelwert von Trefferquote und Genauigkeit zur Beurteilung der Qualität eines Ansatzes gewählt werden. Der Übersichtlichkeit halber wird in diesem Kapitel das F1-Maß für die Ansätze angegeben.

$$F1 = \frac{2 \cdot (precision \cdot recall)}{(precision + recall)} \tag{5.18}$$

Die meisten Algorithmen besitzen Parameter, mit denen ihr Ergebnis beeinflusst werden kann. Um möglichst verlässliche Ergebnisse zu erhalten, wird daher die vorhandene Menge an Testobjekten (im Folgenden Korpus genannt) in zwei Mengen unterteilt. Mit Hilfe der einen wird der Algorithmus trainiert (Trainingskorpus). Das heißt, dass die Parameter und Kategorisierungskonfiguration gefunden wird, mit der auf diesem Korpus das beste Ergebnis (zum Beispiel das höchste F1-Maß) erreicht wird. Mit dieser Konfiguration wird der Algorithmus dann auf den anderen Teil des Korpus (Testkorpus) angewandt. Die hierbei gewonnenen Ergebnisse geben eine Aussage über die Qualität des Ansatzes. Wenn der Korpus jedoch zu klein ist, kann es zu starken statistischen Schwankungen der Ergebnisse kommen. Das bedeutet, dass bei unterschiedlicher Aufteilung des Korpus in Trainings- und Testkorpus ein Algorithmus sehr unterschiedliche Ergebnisse liefert. Um diese Schwankungen zu mildern, wird in diesen Fällen eine sogenannte Kreuzvalidierung eingesetzt. Hierbei wird der Korpus in n Teile geteilt. Jeder dieser Teile wird nun einmal als Testkorpus verwendet, nachdem der Algorithmus mit den jeweils anderen $n-1$ Teilen trainiert wurde. Schließlich wird der Durchschnitt über alle erhaltenen Ergebnisse berechnet und als Ergebnis zurückgeliefert. Dadurch werden Schwankungen, die aufgrund einer zu kleinen Grundgesamtheit entstehen können, gemildert. Wenn zusätzlich eine Stratifikation angewandt wird, wird die Aufteilung in Teilmengen so durchgeführt, dass von jeder Kategorie ungefähr gleich viele Elemente in jeder Teilmenge enthalten sind.

5.4.2 Setting und Korpora

Im Folgenden werden die in dieser Arbeit für die Evaluation der Validierungstechniken eingesetzten Korpora beschrieben. Für die Evaluation des entwickelten Algorithmus ShingleCloud wurden zwei der Korpora eingesetzt - der annotierte TREC-Korpus sowie der METER-Korpus. Zur Evaluation des MiLe-Ansatzes wurden alle beschriebenen Korpora verwendet (siehe Kapitel 6).

Annotierter TREC-Korpus

Der **TREC-Newswire-Korpus** ist eine Sammlung von Zeitungsberichten verschiedener großer amerikanischer Printmedien aus den späten 80er und frühen 90er Jahren. Zu den vertretenen Zeitungen zählen das Wall Street Journal, die Los Angeles Times sowie Artikel der Associated Press und Financial Times.

Der Korpus enthält insgesamt ungefähr 760.000 Dokumente. Zur Evaluation wurde eine kleine annotierte Teilmenge des Korpus verwendet. Diese wurde auch schon von Seo und Croft in [SC08] zum Vergleich einiger der in Kapitel 6.1 beschriebenen Fingerprint-Techniken verwendet. Dieser Korpus besteht aus insgesamt 600 Dokumentenpaaren. Die Paare sind anhand ihrer Ähnlichkeit in insgesamt sechs Kategorien eingeordnet. Die Kategorien ergeben sich aus drei unterschiedenen Stufen der Wiederverwendung in Kombination mit einem asymmetrischen Ähnlichkeitsmaß. Die Ebenen werden durch die Begriffe "most", "considerable" und "partial" umschrieben. Es ergeben sich also folgende Kategorien:

- C1: Most / Most
- C2: Most / Considerable
- C3: Most / Partial
- C4: Considerable / Considerable
- C5: Considerable / Partial
- C6: Partial / Partial

Jedes Dokumentenpaar ist einer dieser Kategorien zugeordnet. Die Einordnung erfolgte durch manuelle Untersuchung und Vergleich des Inhalts der beiden Dokumente eines Paares. Dadurch ergibt sich ein Kategorisierungsproblem. Jeder Wiederverwendungsstufe wird ein prozentualer Ähnlichkeitswert zugeordnet, so kann ein Dokumentenpaar einer bestimmten Kategorie zugeordnet werden. Durch die Verwendung des Abdeckungsmaßes sind jedem Paar zwei Ähnlichkeitswerte zugeordnet, und somit ergibt sich aus der Kombination der jeweils ermittelten Wiederverwendungsstufen die Kategorie. Seo und Croft geben die Schwellwerte für die manuelle Einschätzung der Ähnlichkeit mit 0.80, 0.50 und 0.10 an. Ursprünglich teilten sie den Korpus in einen jeweils 300 Dokumentenpaare umfassenden Trainings- und Testkorpus auf. Der Trainingskorpus wurde verwendet, um die Parameter und Schwellwerte für einen Ansatz zu optimieren. Diese Konfiguration wurde dann auf dem Testkorpus getestet, um die Qualität des Algorithmus zu ermitteln. Durch die geringe Größe des Korpus treten bei einer solchen Herangehensweise jedoch große statistische Schwankungen auf. Aus diesem Grund wird auf den Korpus in dieser Arbeit ein zehnfaches, stratifiziertes Kreuzvalidierungsverfahren zur Evaluation der Ansätze angewandt.

METER-Korpus

Der **METER-Korpus** ([GFW+01, CGPW02]) ist ebenfalls ein journalistischer Textkorpus. Das Akronym "METER" steht dabei für "MEasuring TExt Reuse". Der Korpus wurde also entworfen, um Ansätze zur Messung von Wiederverwendung evaluieren zu können. Er besteht einerseits aus Agenturmeldungen einer großen britischen Presseagentur (British Associated Press) und andererseits aus Zeitungsartikeln von neun verschiedenen britischen Zeitungen (wie zum Beispiel "The Sun", "The Daily Telegraph", "The Times" etc.). Zu jedem Thema einer Agenturmeldung gibt es Zeitungsartikel, die in drei Kategorien eingeteilt sind, abhängig davon, zu welchem Grad die Zeitungsartikel von der Agenturmeldung zu der Geschichte abgeleitet wurden. Die Kategorien heißen demnach *wholly derived*, *partially derived* und *non*

derived; also vollständig, teilweise oder überhaupt nicht abgeleitet. Sie wurden durch manuelle Beurteilung der Agenturmeldungen und Zeitungsartikel durch Journalisten ermittelt. Der Korpus deckt zwei Themengebiete ab. Der Großteil der Artikel befasst sich mit Meldungen über Gerichtsthemen, während es sich bei dem kleineren Teil um Meldungen aus dem Showbusiness-Bereich handelt. Im Gerichtsbereich besitzt der Korpus 770 Artikel, im Showbusiness-Bereich 175. Für die Auswertungen wurden die Artikel aus dem Gerichtsteil des Korpus verwendet. Für jede der drei Kategorien gibt es darin unterschiedlich viele Muster: 166 für non derived, 342 für partially derived und 262 für wholly derived. Dieses Ungleichgewicht muss bei der Auswertung berücksichtigt werden.

Der Korpus weist eine Besonderheit auf: In den meisten Fällen existieren zu einer Geschichte mehrere Versionen einer Agenturmeldung. Dies hängt mit der zeitlichen Entwicklung einer Meldung im Laufe eines Tages ab. Meist werden nach der initialen Meldung eine oder mehrere überarbeitete und umformulierte Versionen herausgegeben, außerdem gibt es oft stark verkürzte Ausgaben einer Meldung, die zum Beispiel in News-Tickern eingesetzt werden können. Schließlich wird abends zumeist eine nochmals überarbeitete und erweiterte Version der Meldung veröffentlicht (die sogenannte "Nightline"). Diese ist in vielen Fällen die Grundlage für die am nächsten Tag in Printmedien erscheinenden Artikel. Oft werden für einen Zeitungsartikel jedoch auch verschiedene Formulierungen aus verschiedenen Versionen einer Agenturmeldung kombiniert - teilweise sogar auf Wortebene. Bei der manuellen Kategorisierung wurde ein Satz, der sich aus verschiedenen Teilen verschiedener Versionen einer Agenturmeldung zu einer Geschichte zusammensetzt, als vollständig abgeleitet kategorisiert. Somit ist aus den Annotationen auch nicht ersichtlich, aus welcher Version einer Agenturmeldung der Artikel abgeleitet wurde. Die Benennung der Kategorien impliziert die Verwendung eines asymmetrischen Ähnlichkeitsmaßes für die Bestimmung der Kategorie. Es soll schließlich bestimmt werden, zu welchem Anteil ein Artikel aus einer Meldung abgeleitet wurde. Dies lässt sich am besten mit Hilfe der Abdeckung der Agenturmeldung durch einen Artikel ausdrücken. Somit kann zur Bestimmung der Kategorie ein einseitiges Abdeckungsmaß verwendet werden, bei dem die Menge der gemeinsamen Elemente durch die Länge des Zeitungsartikels geteilt wird. Dies hat zur Folge, dass die Abdeckung unabhängig von der Länge einer Agenturmeldung ist. Somit können die unterschiedlichen Versionen einer Agenturmeldung konkateniert werden, um einem Algorithmus die Erkennung, ob ein Artikel aus unterschiedlichen Versionen einer Agenturmeldung zusammengesetzt wurde, zu erleichtern.

Weitere Korpora

Der **20-Newsgroups-Korpus** [20n09] ist ein frei verfügbarer Korpus, der die Beiträge 20 verschiedener Newsgroups und insgesamt circa 18800 Dokumente enthält. Durch die gegebene klare Einordnung der Beiträge in bestimmte Themengebiete wird der Korpus häufig dazu verwendet, Verfahren zu testen, die Texte inhaltlich kategorisieren oder Clustern zuweisen. Da Zitierungen und Querverweise in Newsgroups durchaus üblich sind, eignet sich dieser Korpus auch sehr gut für die Erkennung von Wiederverwendung. Es existiert jedoch keine Annotation oder Kategorisierung, so dass die Qualität eines Algorithmus nicht mit Hilfe der oben beschriebenen Maße ermittelt werden kann. Die Qualität eines Algorithmus kann hier anhand der Abweichung seiner Ergebnisse zu denen eines State-of-the-Art-Ansatzes (K-Gram) ermittelt

werden. Zusätzlich können Fälle mit hoher Abweichung noch manuell untersucht werden, um beurteilen zu können, welcher Algorithmus näher an der Wahrheit liegt.

Der **LIS.KOM-Korpus** setzt sich aus verschiedenen Dokumenten zusammen, die während der Evaluation des LIS.KOM-Frameworks (siehe Kapitel 8) gesammelt wurden. Während dieser Evaluation wurden die erfassten Relationen manuell auf ihre Validität hin überprüft und beurteilt. Aus der Menge der beurteilten Relationen wurde der LIS.KOM-Korpus extrahiert. Er dient dazu, den entworfenen MiLe-Algorithmus (siehe Kapitel 6) in dem Anwendungsszenario zu testen, für das er entwickelt wurde. Da es für objektbasierte Dokumente sonst keine frei verfügbaren Korpora gibt, ist dies auch gleichzeitig der einzige Weg, die objektbasierte Version von MiLe zu evaluieren. Er enthält die durch manuelle Beurteilung ausgezeichneten Varianten-Relationen sowie die zugehörigen PowerPoint-Dokumente für die Evaluation des objektbasierten MiLe und Word-Dokumente für die Evaluation des textbasierten MiLe. Auch hier ergibt sich durch die manuell erfolgte Einordnung der Relationen ein Kategorisierungsproblem.

5.5 Evaluation von ShingleCloud

Der ShingleCloud-Algorithmus wurde auf zwei unterschiedlichen Korpora evaluiert: Dem annotierten TREC-Korpus sowie dem METER-Korpus. ShingleCloud wurde dazu entwickelt, automatisiert die Gültigkeit einer gegebenen textbasierten Elementrelation zu bewerten. Hierzu wird der textuelle Inhalt der Relation mit dem Zieldokument der Relation verglichen, um herauszufinden ob, wo und in welchem Umfang er im Zieldokument vorhanden ist. Um den Algorithmus auf den gegebenen Korpora zu evaluieren muss das Abdeckungsmaß (Formel 5.15) berechnet werden, anhand dessen die Zielkategorie eines Dokumentenpaares ermittelt werden kann. Beide genannten Korpora wurden zur Evaluation von Ansätzen zur Messung von Wiederverwendung entwickelt. Zunächst werden die Evaluationsergebnisse auf dem annotierten TREC-Korpus erläutert, bevor in Unterkapitel 5.5.2 auf die Leistung des Algorithmus auf dem METER-Korpus eingegangen wird. Um die Qualität von ShingleCloud beurteilen zu können, wurde der Ansatz auf beiden Korpora jeweils mit N-Gram-Overlap und dem Greedy-String-Tiling-Algorithmus (GST) in unterschiedlichen Konfigurationen verglichen.

5.5.1 Ergebnisse auf dem annotierten TREC-Korpus

Zur Evaluation von ShingleCloud auf dem annotierten TREC-Korpus wurde ein zehnfacher Kreuzvalidierungsansatz auf den Korpus angewendet. Hierbei werden die 600 Dokumente des Korpus in zehn Teile von jeweils 60 Dokumenten zerlegt. Die Anwendungs von Stratifikation bewirkt, dass in jeder Gruppe jede Kategorie im selben Verhältnis wie in der Gesamtmenge vorhanden ist. Da die Teilung des Korpus in zehn Teile zufällig erfolgt und sich dadurch abhängig von der Aufteilung das Ergebnis leicht verändern kann, werden 50 solcher Messungen durchgeführt und das Ergebnis darüber gemittelt. Dadurch wird eine hohe Konfidenz der Ergebnisse erreicht. Die angegebenen Werte liegen mit einer 95 %igen Wahrscheinlichkeit innerhalb der angegebenen Intervalle. Als Klassifikator wurde ein baumbasierter J48-Klassifikator verwendet. Dabei handelt es sich um eine Implementierung des C4.5 Klassifikators ([Qui93, Kot07]). Als Features wurden lediglich die beiden Abdeckungsmaße für ein Dokumentenpaar genutzt. Ein baumbasierter Ansatz eignet sich in diesem Szenario deshalb gut, da ein solcher Klassifika-

tor Entscheidungen häufig aufgrund von Schwellwerten einzelner Features trifft. Dieses Verhalten ist in diesem Fall gewünscht, da der Klassifikator einzig dazu genutzt wird, die optimalen Grenzwerte für das Abdeckungsmaß bezüglich der jeweiligen Kategorien zu finden. Dies spiegelt sich auch in Versuchen mit anderen Klassifikatoren wider, die schwächere Ergebnisse lieferten. Nichtsdestotrotz ist es nicht das Ziel, den optimalen Klassifikator zu finden, sondern die Ansätze selbst zu vergleichen. Hierbei ist lediglich wichtig, dass für alle Ansätze derselbe Klassifikator verwendet wird.

Abbildung 5.1 zeigt die Ergebnisse der Ansätze im Vergleich. Die Notation "SC(x,y,z)" bedeutet, dass der ShingleCloud-Ansatz mit einer Fenstergröße von x, einer Minimallänge von übereinstimmenden Shingles von y und einer Maximallänge für die Unterbrechung einer Übereinstimmung von z konfiguriert wurde.

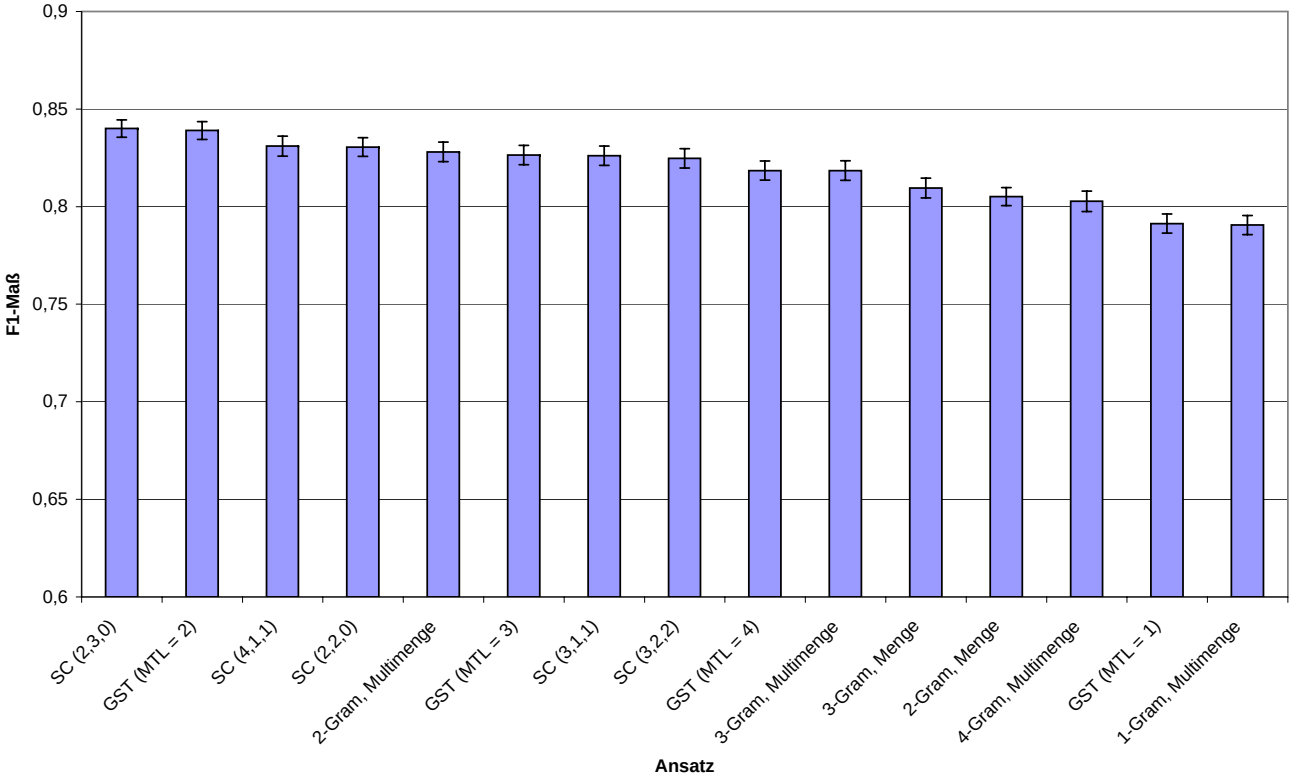

Abbildung 5.1.: Vergleich von String-Matching-Ansätzen auf dem TREC-Korpus

Alle Ansätze bewegen sich qualitativ in einem sehr engen Bereich. Durch die Erweiterungen, zu denen bei ShingleCloud im Vergleich zu N-Gram-Overlap auch die Berücksichtigung und entsprechende Gewichtung der Reihenfolge der Shingles zählt, kann ShingleCloud in der optimalen Konfiguration signifikant bessere Ergebnisse erzielen als N-Gram-Overlap. Bei N-Gram-Overlap zeigt sich zudem, dass die Interpretation der Shingles als Multimenge einen leichten Qualitätsvorteil gegenüber der Mengen-Variante bietet. Der GST-Algorithmus zeigt bei einer minimalen Tile-Länge (MTL) von zwei bis vier auch sehr gute Ergebnisse, die sich nicht signifikant von den ShingleCloud-Ergebnissen unterscheiden. Er weist jedoch im Vergleich ein deutlich schwächeres Laufzeitverhalten auf als die beiden anderen Ansätze.

Die besten Resultate liefert der ShingleCloud-Algorithmus in der Konfiguration mit Fenstergröße zwei, der Mindestlänge einer Übereinstimmung von drei Fenstern und keinerlei Toleranz. Wenn die Fenster-

größe erhöht wird, muss entsprechend die Mindestzahl der Übereinstimmungen gesenkt und ggf. die Anzahl der möglichen Nichtübereinstimmungen erhöht werden. Wie die Abbildung zeigt, gibt es diverse Möglichkeiten, den Algorithmus so zu parametrisieren, dass er gute Ergebnisse auf diesem Korpus erzielt.

5.5.2 Ergebnisse auf dem METER-Korpus

Auch auf dem METER-Korpus werden die drei Ansätze Greedy-String-Tiling, N-Gram-Overlap und ShingleCloud in unterschiedlichen Konfigurationen verglichen. Wie beim TREC-Korpus werden die Ergebnisse mit Hilfe einer zehnfachen Kreuzvalidierung und als Durchschnitt über 50 Durchläufe ermittelt. Auch hier wird der J48-Klassifikator eingesetzt. Als einziges Feature wird die Abdeckung eines Zeitungsartikels durch die zugehörigen Presseagentur-Meldungen (PA-Meldungen) übergeben. Aus den Quelltexten wurden zur Vorverarbeitung Sonderzeichen, Satzzeichen und Stoppwörter entfernt (siehe Anhang A für Ergebnisse mit einer anderen Art der Vorverarbeitung). Tabelle 5.1 zeigt die Ergebnisse der Auswertung. Zum Vergleich ist die Qualität eines naiven Ansatzes als Referenzgüte gegeben. Die Qualität dieses Ansatzes in den einzelnen Kategorien entspricht dem Verhältnis der Muster der jeweiligen Kategorie zur Gesamtmenge. Für die Gesamtgüte wird angenommen, dass alle Dokumentpaare der Kategorie mit den meisten Mustern zugeordnet werden.

Jeder der gezeigten Ansätze schneidet deutlich besser ab als diese Referenzgüte. Es ist auffällig, dass im Gegensatz zum TREC-Korpus der K-Gram-Ansatz mit Fenstergröße eins deutlich bessere Ergebnisse liefert als mit anderen Fenstergrößen, gleiches gilt für GST mit Minimallänge eins. Dies ist jedoch bei Betrachtung der Gegebenheiten einfach zu erklären und liegt an der Art und Weise, wie die Beurteilung der Wiederverwendung bei der Annotation des Korpus erfolgte: Da die Annotationen keine Aussage darüber treffen, von welcher Version einer PA-Meldung ein Artikel abgeleitet ist, wurden alle Versionen einer PA-Meldung konkateniert und dieses PA-Dokument für die Ähnlichkeitsbestimmung verwendet (siehe oben). Ein Zeitungsartikel kann aus diversen einzelnen Tokens verschiedener Versionen einer PA-Meldung zusammengesetzt sein und dennoch als "wholly derived" gelten. Dies hat zur Folge, dass Einzelteile eines abgeleiteten Satzes über das gesamte PA-Dokument verstreut sind. Dies kommt einem Ansatz wie 1-Gram zu Gute, da auf Grund der geringen Fenstergröße jeder dieser verstreuten Tokens gefunden werden kann. Dies hat offenbar einen weit höheren Effekt als die zu erwartenden Verfälschung des Ergebnisses durch irrelevante Übereinstimmungen. Mit den Ergebnissen von 1-Gram und GST mit Minimallänge eins kann ShingleCloud ebenfalls nur dann mithalten, wenn die Fenstergröße klein gewählt wird - die besten Ergebnisse werden daher auch bei ShingleCloud mit Fenstergröße eins erzielt. Auch die Mindestlänge einer Übereinstimmung muss verhältnismäßig klein gewählt werden, um gute Ergebnisse zu erzielen. Wenn sowohl Fenstergröße als auch Mindestlänge etwas größer gewählt werden, muss dementsprechend die maximale Zahl an Unterbrechungen - also die Maximalzahl der Nullen in der Shinge-Cloud - etwas höher gewählt werden (zum Beispiel vier). Alle drei Algorithmen liegen qualitativ recht nah beieinander. ShingleCloud zeigt jedoch in vielen Konfigurationen gute und teilweise bessere Ergebnisse als beide Referenzalgorithmen und zeigt dabei ein deutlich besseres Laufzeitverhalten als GST.

Tabelle 5.1.: Vergleich von String-Matching-Ansätzen auf dem METER-Korpus

Ansatz	Wholly-Derived	Partially-Derived	Non-Derived	Gewichtetes Gesamt-F1-Maß	Konfidenz-intervall
1-Gram	0,7282	0,6393	0,6130	**0,6639**	± 0,0005
2-Gram	0,6823	0,6095	0,5071	**0,6122**	± 0,0011
3-Gram	0,6669	0,6079	0,3974	**0,5826**	± 0,0010
GST (Mindestlänge = 1)	0,6786	0,6580	0,6108	**0,6516**	± 0,0011
GST (Mindestlänge = 2)	0,7141	0,6514	0,5213	**0,6447**	± 0,0013
GST (Mindestlänge = 3)	0,6886	0,6489	0,4911	**0,6284**	± 0,0007
GST (Mindestlänge = 4)	0,6397	0,6119	0,3205	**0,5585**	± 0,0019
ShingleCloud 2,3,4	0,7250	0,6210	0,5462	**0,6403**	± 0,0006
ShingleCloud 1,1,2	0,6600	0,5995	0,5775	**0,6154**	± 0,0010
ShingleCloud 1,2,0	0,6903	0,6722	0,6109	**0,6652**	± 0,0007
ShingleCloud 3,1,3	0,6734	0,6484	0,5051	**0,6260**	± 0,0011
Baseline	0,3402	0,4442	0,2155	**0,4442**	-

5.5.3 Zusammenfassung und Beurteilung

Die Evaluationen haben gezeigt, dass ShingleCloud ein schneller und leistungsfähiger Algorithmus für die Fuzzy-Suche von Strings ist. Auf beiden Korpora ist ShingleCloud entweder genauso gut oder sogar besser als die Vergleichsalgorithmen. Hinzu kommt, dass die Lokalisierung von Übereinstimmungen eines Quelltextes in einem Zieldokument inhärent durch die resultierende ShingleCloud, die sowohl maschinen- als auch menscheninterpretierbar ist, unterstützt wird. Somit bietet ShingleCloud nicht nur die erwünschte Funktionalität für die Validierung textbasierter Elementrelationen, sondern leistet auch im Bereich der Ähnlichkeitssuche von Strings eine Verbesserung gegenüber dem State-of-the-Art.

ShingleCloud im LIS.KOM-Framework

Im LIS.KOM-Framework wird ShingleCloud zur Validierung von Elementrelationen eingesetzt. Zu bestimmten Zeitpunkten wird der Textinhalt einer Relation, also der kopierte Text, mit dem Zieldokument der Relation abgeglichen und so determiniert, ob die Relation noch gültig ist oder nicht mehr (siehe Kapitel 7). Theoretisch kann er aber für verschiedene String-Matching-Aufgaben eingesetzt werden. ShingleCloud weist ein Laufzeitverhalten der Ordnung $O(n)$ auf. Allerdings kann er, gegeben durch seine Struktur, nicht in Zusammenhang mit performanten Datenstrukturen eingesetzt werden, die erforderlich sind, wenn sehr viele Dokumente miteinander verglichen werden sollen.

Für die Validierung einer Relation im LIS.KOM-Framework wird kein Abdeckungsmaß berechnet. Eine Relation gilt dann als valide, wenn eine Sequenz der Minimallänge der Nadel im Heuhaufen gefunden wurde. Es genügt also, wenn ShingleCloud bei der Suche nach dem kopierten Relationstext im Zieldokument eine den gegebenen Parametern entsprechende Übereinstimmung findet. Die Konfiguration, mit der ShingleCloud für die Validierung der Relationen eingesetzt wurde, musste daher, um fälschlich als gültig eingestufte Relationen zu vermeiden, strenger gewählt werden als die Konfiguration, die auf dem TREC- bzw. METER-Korpus die besten Ergebnisse erreicht hat. Es wurde ein ShingleCloud mit Fenstergröße drei, Minimallänge einer Übereinstimmung von fünf und einer maximalen Anzahl von Nullen von zwei zur Validierung eingesetzt. Dies bedeutet, dass eine Relation nur dann als gültig gewertet wird, wenn aus dem kopierten Text der Elementrelation mindestens sieben aufeinanderfolgende Tokens noch im Zieldokument vorhanden sind. Dies reicht aus, um die Relation als gültig zu bewerten, auch wenn der ursprünglich kopierte Text deutlich länger ist.

Weitere Anwendungsmöglichkeiten

Die Evaluationen haben gezeigt, dass ShingleCloud in vielen Szenarien als String-Matching-Algorithmus eingesetzt werden kann. Ein mögliches Einsatzsszenario ist das Auffinden von Plagiaten. Der Algorithmus wurde für einen Anwendungsfall entwickelt, in denen eine Nadel mit einem Heuhaufen verglichen werden soll. Der Algorithmus soll dabei erkennen ob, wo und in welchem Umfang die Nadel im Heuhaufen vorhanden ist. In einem solchen Fall ist der Aufbau eines invertierten Index meist weniger sinnvoll, und das günstige Laufzeitverhalten des ShingleCloud-Ansatzes kommt zum Tragen.

Ein Anwendungsszenario, außerhalb des LIS.KOM-Frameworks, in dem ShingleCloud bereits eingesetzt wurde, ist das Holinshed-Projekt [Hol09]. Bei den Holinshed-Chroniken handelt es sich um ein Werk britischer Geschichte aus dem 16. Jahrhundert, zu dem diverse Autoren unterschiedlicher Konfession beitrugen. Die Chroniken dienten vielen bekannten Autoren, darunter William Shakespeare, als Quelle und geschichtliche Grundlage. Sie wurden in zwei Versionen herausgegeben. Die Originalversion wurde 1577 veröffentlicht. 1587 erfolgte die Veröffentlichung einer Revision, die an vielen Stellen geändert, ergänzt oder umgestellt worden war. Beide Ausgaben liegen in digitaler, im TEI-Format [Ini09] auf Paragraphenebene ausgezeichneter Form vor. Jede Ausgabe besteht hierbei aus mehr als 20.000 Paragraphen. Eine Aufgabe des Holinshed-Projektes bestand darin, herauszufinden, ob, inwieweit und wo die Paragraphen der ursprünglichen Version in der Revision der Chronik enthalten sind. ShingleCloud wurde hier eingesetzt, um einem Experten zu jedem Paragraphen des Ursprungstextes die entsprechenden Übereinstimmungen in der Revision zu visualisieren. Hierzu wurde im Rahmen des Projekts von Cummings und Mittelbach der ShingleCloud-Algorithmus erweitert und ein entsprechendes Frontend entwickelt (siehe [CM09a, CM09b]). Mit Unterstützung dieses Werkzeugs entstand ein Annotations-Set, das jedem Paragraphen im Ursprungstext, wenn existent, den entsprechenden Abschnitt oder die entsprechenden Abschnitte im Zieltext zuweist.

6 Validierung von Variantenrelationen mit Hilfe von Fingerprinting-Techniken

Mit Hilfe der in Kapitel 5 vorgestellten Techniken können auch Variantenrelationen textbasierter Dokumente validiert werden. Während es für Elementrelationen jedoch ausreichend ist, den Inhalt eines Dokuments mit dem in der Relation gespeicherten wiederverwendeten Text abzugleichen, ist es für die Validierung und Überprüfung von Variantenrelationen mit den oben beschriebenen Techniken notwendig, dass die Inhalte beider an der Relation beteiligten Dokumente zugreifbar sind. Dies ist in dem hier gegebenen Szenario aber oft nicht der Fall. Die Vorhaltung der Dokumente selbst ist in dem konzipierten System nicht grundsätzlich vorgesehen und oftmals auch nicht gewünscht, wenn Dokumente beispielsweise vertraulich sind. Mit Hilfe von Fingerprinting-Techniken ist es jedoch möglich, eine Repräsentation eines Dokuments zu erzeugen, die als Attribut einer Relation gespeichert werden kann und anhand derer es möglich ist, eine Variantenrelation zu validieren, ohne Zugriff auf die beteiligten Dokumente zu haben. Darüber hinaus soll eine solche Technik es ermöglichen, die Positionen der inhaltlichen Übereinstimmungen innerhalb eines Dokumentes anhand seines Fingerprints zu bestimmen. Das Problem der Validierung einer Variantenrelation ist auf die Erkennung von Übereinstimmungen zwischen den durch die Relation verbundenen Dokumenten abbildbar. Der Grad der Übereinstimmung zweier Dokumente kann mit Hilfe von Fingerprinting-Techniken bestimmt werden.

Im Folgenden werden verschiedene Fingerprinting-Techniken beschrieben und auf ihre Anwendbarkeit in dem gegebenen Szenario hin geprüft. Schließlich wird der hierfür entworfene MiLe-Algorithmus vorgestellt und auf verschiedenen Korpora mit den anderen Techniken verglichen. Zudem wird eine Anpassung des Algorithmus auf objektbasierte Dokumente, wie PowerPoint-Präsentationen, vorgestellt und ebenfalls evaluiert.

6.1 Existierende Fingerprinting-Ansätze

Man kann bei Fingerprinting-Ansätzen zwei grundlegende Richtungen unterscheiden.

Ansätze zur **Erkennung von Nahezu-Duplikaten**: Diese Ansätze generieren vergleichsweise kurze Fingerprints konstanter Länge wobei ein Fingerprint pro Dokument erzeugt wird. Dabei handelt es sich prinzipiell um Hashwerte. Im Gegensatz zu gewöhnlichen Hashfunktionen (wie MD5), sind diese jedoch ähnlichkeitssensitiv. Das heißt, dass für eine geringfügige Änderung an einem Dokument nahezu derselbe Hashwert für das Dokument erzeugt wird wie vor der Änderung. Um die Ähnlichkeit zweier solcher Fingerprints zu bestimmen, werden oftmals Distanzmaße wie die Hamming-Distanz (siehe Kapitel 5.1.3) verwendet. Solche Techniken liefern jedoch aufgrund des stark komprimierten Wertebereichs nur verlässliche Ergebnisse, wenn sich die verglichenen Dokumente sehr ähnlich sind. Zu diesen Ansätzen zählen die Fingerprint-Techniken von Charikar [Cha02], Broder [Bro00] und Henzinger, die Elemente aus beiden Algorithmen kombiniert [Hen06]. Da eine Validierung von Variantenrelationen nicht nur im Nahezu-Duplikat-Bereich erfolgen soll, werden diese Ansätze im Folgenden nicht weiter betrachtet.

Ansätze zur **Erkennung lokaler Wiederverwendung**: Diese Ansätze erlauben auch eine Erkennung von Wiederverwendung einzelner Textstellen und sind in einem großen Ähnlichkeitsbereich einsetzbar. Die durch sie generierten Fingerprints sind von der Länge des zu Grunde liegenden Dokuments abhängig. Für ein Dokument ergibt sich dabei meist eine Menge von Fingerprints, die für die Ähnlichkeitsberechnung herangezogen werden. Verschiedene existierende Ansätze dieser Art werden im Folgenden beschrieben.

6.1.1 K-Gram

K-Gram ist im Prinzip identisch mit dem in Kapitel 5.2 vorgestellten N-Gram-Overlap. Bei diesem Ansatz werden die Hashwerte der extrahierten n-Gramme als Fingerprints des Dokuments gespeichert. Hierbei können diese wiederum als Menge oder Multimenge behandelt werden. Typischerweise werden 32bit Hashwerte der n-Gramme gespeichert, so dass sich für die Gesamtlänge der resultierenden Fingerprints in Bytes Folgendes ergibt, falls alle n-Gramme des Dokuments unterschiedlich sind:

$$M(D) = (|D| - k + 1) * 4\ Bytes \tag{6.1}$$

Die Ähnlichkeit zwischen zwei Dokumenten wird mit Hilfe des Jaccard-Maßes berechnet. Die Anzahl der von K-Gram erzeugten Fingerprints entspricht ungefähr der Anzahl der Tokens in einem Dokument. Dementsprechend hoch ist der Speicherbedarf. Die Qualität der Ergebnisse ist identisch mit der von N-Gram-Overlap und einige der im Folgenden beschriebenen Algorithmen basieren auf K-Gram. Weiterhin dient K-Gram, da der Ansatz nicht komprimiert, als qualitativer Maßstab für die weiteren Ansätze.

0-mod-p

Anstatt alle n-Gramme zu verwenden, wie es in K-Gram der Fall ist, wählt der 0-mod-p-Algorithmus [Man94] eine Untermenge der Fingerprints, die K-Gram liefert, aus. Eine Anforderung hierbei ist, dass für denselben Text immer dieselben Hashwerte ausgewählt werden. Bei einer zufälligen Auswahl ist dies nicht gewährleistet. Deshalb wählt der gegebene Algorithmus seinem Namen entsprechend nur die Hashwerte $H(k)$ aus, die folgende Bedingung erfüllen:

$$H(k) \mod p \equiv 0 \tag{6.2}$$

Es werden also alle Hashwerte ausgewählt, für die sich eine restlose Division durch p durchführen lässt. Je größer p ist, desto weniger Hashwerte werden für den Fingerprint ausgewählt und desto ungenauer wird der Algorithmus. Die Anzahl der ausgewählten Hashwerte ist im Durchschnitt die Anzahl der von K-Gram ausgewählten Hashwerte geteilt durch p. Für den Parameter p werden typischerweise Werte zwischen vier und acht gewählt. Ein Nachteil dieses Algorithmus ist die Abdeckung eines Textes mit den durch 0-mod-p extrahierten Fingerprints. Bedingt durch die Verteilung der Hashwerte kann es vorkom-

men, dass viele Hashwerte in kurzen Abständen ausgewählt werden, während an anderen Stellen für längere Sequenzen kein Hashwert gewählt wird.

6.1.2 Winnowing

Winnowing ist ein weiterer Ansatz, der aus der Multimenge der von K-Gram erzeugten Hashwerte eine Untermenge an Fingerprints auswählt. Er wurde von Saul Schleimer in [SWA03] vorgestellt. Das Ziel dieses Ansatzes ist es eine ähnliche Anzahl an Hashwerten auszuwählen wie 0-mod-p, dabei aber eine bessere Abdeckung zu gewährleisten. Dafür wird über die Hashwerte der aus dem Text extrahierten n-Gramme ein weiteres, im Verhältnis zu n deutlich größeres, Fenster - das Winnowing-Fenster - geschoben. Aus diesem Fenster wird jeweils der geringwertigste Hashwert gewählt und in die Menge der Fingerprints aufgenommen. Sollte es mehrere gleiche Hashwerte mit geringstem Wert geben, wird der am weitesten rechts im Fenster ausgewählt. Ein und derselbe Hashwert, der in einem vorherigen Fenster bereits ausgewählt wurde, wird nicht erneut ausgewählt. Das folgende Beispiel verdeutlicht die Vorgehensweise des Winnowing-Algorithmus.

> Dies ist ein Text, der als Beispiel dient, ein Text. Dies ist ein Text so ist ein Text.
> *(a) Eingangstext*
>
> [Dies ist], [ist ein], [ein Text], [Text der], [der als], [als Beispiel], [Beispiel dient]
> [dient ein], [ein Text], [Text dies], [dies ist], [ist ein], [ein Text], [Text so], [so ist]
> [ist ein], [ein Text]
> *(b) 2-Gramme des Eingangstextes*
>
> 12, 23, 6, 43, 17, 11, 26, 38, 6, 31, 12, 23, 6, 43, 5, 23, 6
> *(c) Hypothetische Hashwerte der 2-Gramme*
>
> {12, 23, ***6***, 43} {23, 6, 43, 17} {6, 43, 17, 11} {43, 17, ***11***, 26} {17, 11, 26, 38}
> {11, 26, 38, ***6***} {26, 38, 6, 31} {38, 6, 31, 12} {6, 31, 12, 23} {31, 12, 23, ***6***}
> {12, 23, 6, 43} {23, 6, 43, ***5***} {6, 43, 5, 23} {43, 5, 23, 6}
> *(d) 4-Gramme der Hashwerte durch Winnowing*
>
> 6 11 6 6 5
> *(e) Durch Winnowing ausgewählter Fingerprint*

Eine weitere Variante diese Algorithmus - "robustes Winnowing" - sieht als Abwandlung vor, dass, wenn es zwei gleiche Hashwerte mit niedrigstem Wert in einem Fenster gibt, derselbe Hashwert wie im Fenster eine Position weiter links gewählt wird. Schleimers Experimente zeigen, dass Winnowing bessere Ergebnisse liefert als 0-mod-p [SWA03]. Er gibt eine untere Schranke der durch den Algorithmus ausgewählten Fingerprints mit

$$N_{winnowing} = \frac{2}{w+1} N_{K-Gram} \tag{6.3}$$

Im Vergleich zu 0-mod-p ist Winnowing mit höherer Auflösung parametrisierbar. Durch Winnowing wird garantiert, dass, wenn zwei Dokumente eine übereinstimmende Sequenz besitzen, die mindestens die Länge des Winnowing-Fensters haben, mindestens ein Token aus dieser Sequenz ausgewählt wird.

6.1.3 Hailstorm

Hailstorm [HBCH09] arbeitet nach einem ähnlichen Prinzip wie Winnowing, erlaubt aber eine noch bessere Abdeckung der Dokumente. Während die Auswahl eines Shingles in Winnowing lediglich von den Hashwerten im selben Winnowing-Fenster abhängt, ist die Auswahl der Hashwerte in Hailstorm wie auch bei 0-mod-p global. Zusätzlich zu den durch K-Gram erzeugten Hashwerten, wird bei Hailstorm pro Token des Dokuments ein weiterer Hashwert erzeugt. Anhand der Hashwerte der Tokens wird entscheiden, ob der Hashwert eines Shingles in die Menge der Fingerprints aufgenommen wird. Es wird jedes Shingle S gewählt, für das der minimale Hashwert aller Tokens innerhalb S entweder an erster oder letzter Position innerhalb von S steht. Dadurch ist für jeden Token - außer den ersten und letzten $k-1$ Tokens des Dokuments - gewährleistet, dass er in mindestens einem der gewählten Shingles vertreten ist. Durch Hailstorm werden im Schnitt ungefähr $\frac{2}{k}$ Shingles ausgewählt. Auch dieser Algorithmus garantiert, dass wenn zwei Dokumente eine genügend lange, übereinstimmende Sequenz besitzen, aus dieser Sequenz ein Shingle für den Fingerprint ausgewählt wird. Das Kompressionsverhalten von Hailstorm ist allein abhängig von der Shingle-Größe k. Dies hat zur Folge, dass man große Fenstergrößen braucht, um entsprechende Kompression zu erwirken. Für den Anwendungsfall, für den Hailstorm entwickelt wurde, nämlich die Erkennung des Ursprungs eines Webdokuments sind diese gut verwendbar, da in diesem Szenario meist längere Sequenzen von Text unverändert übernommen werden. Eine Fenstergröße von 8, wie sie sich in [HBCH09] als gut herausgestellt hat, erwirkt hierbei eine Kompression von 75 %. Für die Erkennung von Wiederverwendung in Pressetexten oder die Erkennung von Plagiaten, wie auch für die Validierung von Relationen sind solche Fenstergrößen weniger geeignet, da der Algorithmus dann sehr anfällig gegenüber kleinen Änderungen ist.

6.1.4 Hashbreaking

Während die vorhergehenden Ansätze darauf beruhen, eine Auswahl aus der von K-Gram erzeugten Menge an Hashwerten zu treffen, funktioniert Hashbreaking [BDGM95] auf andere Art und Weise. Der Ansatz zählt zu den nicht überlappenden Methoden, das heißt, es werden keine Shingles aus dem Text erzeugt. Vielmehr wird der Text in nicht überlappende Teile zerlegt, für die jeweils ein Hashwert berechnet wird. Um die Abdeckung zweier Dokumente zu berechnen werden diese Hashwerte dann verglichen. Eine Möglichkeit der Unterteilung, ist eine Unterteilung anhand von Satzzeichen. So könnte ein Text beispielsweise in einzelne Sätze oder Nebensätze zerlegt werden, für die dann Hashwerte berechnet und verglichen werden können. Dies hat sich aber, insbesondere für webbasierte Dokumente, als wenig sinnvoll erwiesen [SC08]. Deshalb wird ein Text beim Hashbreaking anhand eines anderen Kriteriums zerlegt: Zunächst werden für alle Tokens des Textes Hashwerte erzeugt. Dann wird - wie bei 0-mod-p - berechnet, welche der Hashwerte bei einer Division durch p Null ergeben. Diese Terme markieren dann die Stellen im Text, an denen der Text geteilt wird. Für die Teilstücke wird dann wiederum ein Hashwert

berechnet und in die Menge der Fingerprints aufgenommen. Auch in dieser Form offenbart der Ansatz jedoch Schwächen. Ähnlich wie 0-mod-p können sich sehr kurze und sehr lange Sequenzen ergeben. Insbesondere sehr kurze Sequenzen, die sehr häufige Worte enthalten verzerren hierbei das Ergebnis einer Ähnlichkeitsberechnung. Um dem entgegenzuwirken modifizieren Seo und Croft in [SC08] Hashbreaking so, dass Sequenzen, die kürzer als p sind, ignoriert werden. Durch Hashbreaking - insbesondere wenn die genannte Modifikation eingesetzt wird - werden vergleichsweise wenige Hashwerte in den Fingerprint des Dokuments aufgenommen, dafür ist der Algorithmus sehr anfällig gegenüber kleinen Änderungen im Text. Ändert sich in einer gewählten Sequenz nur ein Buchstabe, gilt die ganze Sequenz als nicht übereinstimmend mit der Originalsequenz.

6.1.5 Diskrete Kosinustransformation (DCT)

Die Diskrete Kosinustransformation [ANR74] ist eine reellwertige Version der Diskreten Fouriertransformation. Mit ihrer Hilfe wird ein zeitbasiertes Signal in ein frequenzbasiertes transformiert. Eine Eigenschaft ist hierbei, dass bei der Transformation entstehende Koeffizienten für hohe Frequenzen mehr Gewicht haben als für niedrige Frequenzen. Diese Eigenschaft wird in der Bildverarbeitung beispielsweise zur Komprimierung genutzt (z.B. bei JPEG [Ste00]), kann aber auch als robuste Hashfunktion zur Authentisierung von Bildern eingesetzt werden (siehe [LfC97]). Seo und Croft nutzen diese Eigenschaft der DCT in [SC08] um die Änderungsanfälligkeit des beschriebenen Hashbreaking-Algorithmus zu vermindern. Hierzu wird der Text mit Hilfe des Hashbreaking-Ansatzes in einzelne Sequenzen unterteilt. Darauf wird ein Hashwert für jeden Token einer Sequenz berechnet. Die Hashwerte einer Sequenz werden nun vertikal verschoben, so dass der Median der Hashwerte bei Null liegt. Daraufhin wird noch auf den maximalen Hashwert der Sequenz normalisiert. Die sich daraus ergebenden Werte für eine Sequenz können als Zeitsignal interpretiert werden. Dieses kann dann mit Hilfe der diskreten Kosinustransformation in den Frequenzbereich transformiert werden:

$$X_k = \sum_{n=0}^{N-1} x_n \cos\left[\frac{\pi}{N}\left(n+\frac{1}{2}\right)k\right] \qquad k = 0,\ldots,N-1. \tag{6.4}$$

Die hierzu verwendete Formel ist die am häufigsten gebräuchliche Version der DCT und eine leichte Abwandlung des Originals aus [ANR74]. Die sich ergebenden Koeffizienten werden nun auf eine Länge von wenigen Bit (2,3 oder 4) quantifiziert, so dass sich für jede Sequenz ein Hashwert der Länge 32 Bit ergibt. Dieser geht dann in die Menge der Fingerprints des Dokuments ein. Welche der Koeffizienten in den jeweiligen Fingerprint eingehen hängt auch von der Länge der Textsegment ab. Seo und Croft schlagen vor, nur die p Koeffizienten der niedrigsten Frequenzen aufzunehmen, wenn $N > p$. Hierbei ist p der Parameter, der verwendet wird um den Text mittels Hashbreaking in einzelne Segmente zu unterteilen. Die Verteilung der einzelnen Koeffizienten auf den Fingerprint erfolgt wie in Abbildung 6.1 gezeigt.

Die ersten 16 Bit nimmt hierbei der Hashwert des ersten Tokens der Sequenz ein. Die verbleibenden 16 Bit werden dann unter den berechneten Koeffizienten des Frequenzbereichs aufgeteilt. Bei einer

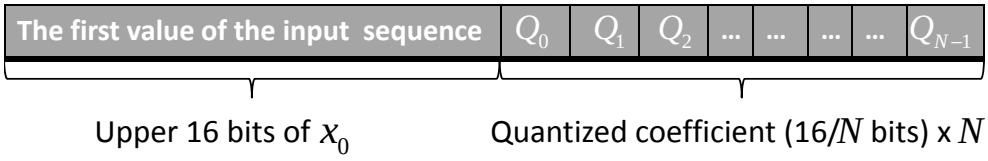

Abbildung 6.1.: Zusammensetzung eines DCT Fingerprints aus [SC08]

gewählten Quantifizierung auf zwei Bit werden also acht Koeffizienten in den Hashwert der Sequenz aufgenommen. Der Ansatz zeigt eine gewisse Robustheit gegenüber Änderungen in der Mitte oder am Ende einer Sequenz. Eine Änderung am Anfang eines Segments führt jedoch wie bei Hashbreaking auch zu einer Änderung des kompletten Fingerprints der Sequenz, unabhängig davon, wie weit der Rest der Segmente übereinstimmt.

6.1.6 Beurteilung

K-Gram ist die Basis für die meisten der anderen Algorithmen und liefert in diversen Evaluationen gute Ergebnisse - zumindest im Vergleich zu den anderen hier vorgestellten Ansätzen. Deswegen wird er in vielen Fällen auch als Bezugsgröße zur Messung der Qualität von Fingerprinting-Ansätzen verwendet (zum Beispiel in [SC08], [HBCH09]). Allerdings muss bei K-Gram auch eine große Anzahl großer Fingerprints gespeichert werden, was abhängig von der für die K-Gram-Fingerprints gewählt Bitlänge, dazu führen kann, dass die Fingerprints mehr Speicher verbrauchen als das Dokument selbst.

Winnowing, **Hailstorm** und **0-mod-p** basieren allesamt auf K-Gram und wählen eine Untermenge der Fingerprints. Sie unterscheiden sich durch die Abdeckung und die globale Unabhängigkeit voneinander und erzielen in allen Evaluationen deutlich schwächere Ergebnisse als K-Gram. Der Vorteil dieser Ansätze ist der geringere Speicherverbrauch durch die Reduzierung der vorgehaltenen Fingerprints sowie der damit verbundene Geschwindigkeitsgewinn bei der Auswertung der Fingerprints.

Hash-Breaking wählt in der von Seo und Croft vorgeschlagenen Version nur eine sehr geringe Anzahl von Fingerprints aus, ist aber auch sehr anfällig gegenüber kleinen Änderungen in den Dokumenten. **DCT-Fingerprinting** erhöht die Robustheit von Hash-Breaking durch Einsatz der Diskreten Kosinus Transformation und erzielt dabei bessere Ergebnisse als Hash-Breaking.

Insgesamt gibt es einen offensichtlichen Trade-off zwischen Qualität, Speicherbedarf und Performance. Da die Performance in dem hier gegebenen Szenario zweitrangig ist, soll ein Fingerprint zur Validierung von Variantenrelationen hauptsächlich gute Ergebnisse liefern, wenig Speicherplatz benötigen und den Vergleich zweier Dokumente und damit die Validierung in Echtzeit durchführen können. K-Gram zeigt in den gegebenen Evaluationen zwar die besten Ergebnisse im Vergleich zu anderen existierenden Fingerprinting-Techniken, braucht aber vergleichsweise viel Speicherplatz, außerdem ist ein Algorithmus, der bessere Ergebnisse als K-Gram liefert, dennoch erstrebenswert.

6.2 Der MiLe Fingerprinting-Ansatz

Jeder der in Kapitel 6.1 beschriebenen Ansätze überführt ein Dokument in eine ungeordnete (Multi-) Menge von Fingerprints. Die Fingerprint-Mengen zweier Dokumente werden dann verglichen, um eine Abdeckung bzw. Ähnlichkeit zu berechnen. Die Ordnung der Tokens innerhalb eines Dokuments wird nur auf Ebene einzelner Fingerprints in Form von Segmenten oder Fenstern über eine geordnete Sequenz von Tokens berücksichtigt. Die Menge der Fingerprints eines Dokumentes selbst besitzt jedoch keine Ordnung. Diese Information wird von den bestehenden Ansätzen komplett vernachlässigt. Im Folgenden wird eine Fingerprint-Technik beschrieben, die einem Dokument einen von der Länge des Dokumentes abhängigen Fingerprint in Form eines Bitstrings zuweist. Dabei werden die Zusammenhänge der Tokens innerhalb des Dokuments und deren Ordnung berücksichtigt. Durch die Berücksichtigung der Ordnung lässt sich nicht nur eine genauere Repräsentation der Ähnlichkeit zweier Dokumente erreichen, sie ermöglicht es auch, anhand der Übereinstimmungen der Fingerprints die Positionen der Übereinstimmungen im Text zu bestimmen. Die Grundversion des MiLe-Fingerprints ist für textbasierte Dokumente geeignet. Mit Hilfe einer Erweiterung lässt sich der Fingerprint jedoch auch zur Validierung der Variantenrelationen für objektbasierte Dokumente, wie zum Beispiel PowerPoint-Präsentationen, einsetzen.

6.2.1 MiLe für textbasierte Dokumente

Der Algorithmus zum Vergleich zweier textbasierter Dokumente mit Hilfe von MiLe setzt sich wie bei allen Fingerprint-Techniken aus zwei Hauptschritten zusammen: Zunächst wird der Fingerprint für die Dokumente gebildet, bevor aus den Fingerprints von zwei Dokumenten die Ähnlichkeit der Dokumente berechnet werden kann.

Berechnung des Fingerprints

Die Erzeugung eines MiLe-Fingerprints für ein Dokument läuft dabei folgendermaßen ab:

1. Der Text des Dokuments wird extrahiert und vorverarbeitet. Die Vorverarbeitung kann Säuberung, Anpassung der Groß/Kleinschreibung, Stoppwortfilterung und Stammbildung bzw. Lemmatisierung umfassen.

2. Aus dem vorverarbeiteten Text werden nicht überlappende Tokens auf Wortebene gebildet.

3. Jeder Token wird quantifiziert und auf einen Hashwert mit geringer Anzahl an Bits abgebildet (z.B. 4 Bit). Dies kann zum Beispiel durch Bildung eines MD5-Hashwertes und Verwendung der ersten n Bits erfolgen, in laufzeitkritischen Anwendungen, kann jedoch auch eine effizientere Methode zur Quantisierung angewendet werden.

4. Die Bitwerte werden in der Reihenfolge der Tokens im Text hintereinander in einen Bitstring geschrieben. Dieser Bitstring enthält schließlich den MiLe-Fingerprint des Dokuments.

MiLe zeichnet sich durch vergleichsweise kurze Fingerprints aus. Der resultierende Fingerprint kann als verlustbehaftete und stark komprimierte Version des Textes selbst betrachtet werden. Jeder Token des

Textes wird in einen Abschnitt des MiLe-Fingerprints überführt, wobei die Reihenfolge der Tokens aus dem Originaltext erhalten bleibt. Wenn also bei der Vorverarbeitung keine Stoppwortlisten eingesetzt werden, kann anhand eines MiLe-Fingerprints die exakte Position einer Überschneidung innerhalb der verglichenen Texte bestimmt werden. Durch die geringe Bitlänge, die für jeden Token im Text verwendet wird ist eine sehr große Kollisionswahrscheinlichkeit gegeben. Deshalb müssen bei der Bestimmung der Ähnlichkeit die Tokens in ihrem Zusammenhang betrachtet werden. Während alle anderen Ansätze die resultierenden Fingerprints als ungeordnete Menge betrachten und somit die Ordnung der Fingerprints für die Berechnung der Ähnlichkeit keine Rolle spielt, wird diese Information bei der Bestimmung der Ähnlichkeit zweier MiLe-Fingerprints miteinbezogen.

Ähnlichkeitsbestimmung

Zur Ähnlichkeitsbestimmung werden die gegebenen Fingerprints wie Strings behandelt, zwischen denen Übereinstimmungen gefunden werden sollen. Dies läuft in folgenden Schritten ab:

1. Zunächst werden Shingles aus den Fingerprints zweier Dokumente extrahiert. Die Shingles es ersten Fingerprints werden hierbei in ihrer Reihenfolge in einer einfachen Liste gespeichert. Die Shingles des zweiten Fingerprints werden in einer Hashtabelle gemeinsam mit einem Zähler für die Häufigkeit innerhalb des Fingerprints abgelegt. Ein Token entspricht hierbei der gewählten Bitlänge für den Fingerprint. Die gewählte Shingle-Größe entspricht der Minimallänge m, die eine Übereinstimmung zwischen den Dokumenten mindestens haben muss. Durch diesen Parameter wird gleichzeitig auch die Wahrscheinlichkeit einer Kollision erheblich gemindert. Bei einer Bitlänge von vier und einer Shingle-Größe von fünf ergibt sich eine für Kollisionen relevante Gesamtbitlänge von 20 Bit. Die Kollisionswahrscheinlichkeit wird im weiteren Verlauf des Kapitels betrachtet.
2. Ein Integer, der die Übereinstimmungen zwischen den Fingerprints zählt, wird initialisiert.
3. Die in der Liste gespeicherten Shingles werden durchlaufen und jeweils mit Hashtabelle verglichen.
4. Ist das Shingle in der Hashtabelle enthalten, wird abhängig vom letzten Shingle der Wert Eins oder m zum Zähler hinzugezählt. Wenn das vorherige Shingle ebenfalls in der Hashtabelle enthalten war, wird lediglich eins hinzugezählt, da dies bedeutet, dass zwei Shingles in Folge übereinstimmen (also $m+1$ Tokens). Ist das vorherige Shingle nicht in der Hashtabelle enthalten, bedeutet dies, dass erneut m Tokens in Folge übereinstimmen, weshalb m zum Zähler addiert wird. Ist der Wert enthalten wird der Zähler in der Hashtabelle, um doppelte Übereinstimmungen zu vermeiden, um eins vermindert.
5. Um die Abdeckung des einen Fingerprints durch den anderen und somit die Abdeckung des einen Dokuments durch das andere zu berechnen, wird schließlich, wenn alle Shingles durchlaufen sind, der Zähler durch die Länge des jeweiligen Fingerprints dividiert.

Für die Abdeckung zweier Dokumente A und B ergibt sich mit Zählergröße c:

$$S_{MiLe}(A,B) = \frac{c}{N_A - m + 1} \tag{6.5}$$

Die Abdeckung zweier Fingerprints und somit zweier Dokumente wird also nicht auf Basis der Anzahl übereinstimmender Shingles, sondern auf Basis übereinstimmender Tokens (also Wörter) durchgeführt. Dies führt zu einer erhöhten Genauigkeit der Berechnung der Ähnlichkeit. Die Gesamtzahl der Shingles in einem Text ist immer ungefähr gleichgroß wie die Gesamtzahl der Tokens. Der Divisor für die Berechnung der Abdeckung ist also in beiden Fällen ungefähr gleich groß. Der Dividend kann sich in den beiden Fällen - abhängig von der Fenstergröße und der Verteilung der Übereinstimmungen im Text - stark unterscheiden. Dies führt zu einer verfälschten Berechnung der Abdeckung, wenn die Abdeckung auf Basis übereinstimmender Shingles berechnet wird.

Parametrisierung

Die zwei dominierenden Parameter des Ansatzes sind die gewählte Bitlänge sowie die Minimallänge. Ein MiLe-Fingerprint mit gewählter Bitlänge von b für ein Dokument D hat eine Länge von $|D| \cdot b/8$ Byte. Man kann die Länge des Fingerprints also über die gewählte Bitlänge der einzelnen Hashwerte bei der Erzeugung des Fingerprints beeinflussen. Wenn eine sehr geringe Bitlänge gewählt wurde (z.B. 2 Bit) muss für die Ähnlichkeitsbestimmung eine große Minimallänge vorausgesetzt werden, um die Wahrscheinlichkeit von Kollisionen zu verringern, dadurch wird der Algorithmus strenger, da längere Sequenzen von Übereinstimmungen vorausgesetzt werden. Wurde eine größere Bitlänge bei der Erzeugung des Fingerprints gewählt, kann mit kürzeren Minimallängen gearbeitet werden. Da die Bitlänge bei der Erzeugung eines Fingerprints festgelegt wird, kann sie im Nachhinein nicht mehr angepasst werden. Zwei zu vergleichende Fingerprints müssen mit derselben Bitlänge erzeugt worden sein, um vergleichbar zu sein. Die Minimallänge einer Übereinstimmung hingegen wird erst bei der Ähnlichkeitsberechnung eingesetzt und kann somit auch nach Erzeugung der Fingerprints noch angepasst werden. So kann auch nach Erzeugung der Fingerprints das Verhalten des Algorithmus auf den Anwendungsfall angepasst werden.

Durch die beiden Parameter Bitlänge und Minimallänge der Übereinstimmungen ist es möglich den Algorithmus sehr flexibel zu parametrisieren. Eine Besonderheit von MiLe zeigt sich bei Betrachtung der Wahrscheinlichkeit, dass zwei Sequenzen als Übereinstimmung klassifiziert werden. Wenn beide Sequenzen die Minimallänge aufweisen und vollständig übereinstimmen, ergibt sich selbstverständlich eine Wahrscheinlichkeit von 1 dafür, dass die Übereinstimmung erkannt wird. Dies ist auch bei den anderen Algorithmen der Fall. Bei K-Gram wird dann eine Übereinstimmung erkannt, wenn die zwei Sequenzen in mindestens k Tokens gleich sind. Im umgekehrten Fall, wenn sich die beiden verglichenen Sequenzen, im Fall von K-Gram zwei Shingles und bei MiLe zwei Sequenzen der Minimallänge m, in jedem Token unterscheiden, wird dies nur bei einer Kollision der Hashwerte als Übereinstimmung erkannt. Die Wahrscheinlichkeit hierfür ist:

$$P_{acc} = \frac{1}{2^d} \tag{6.6}$$

Hierbei ist d die Bitlänge des gewählten Hashwertes. Im Fall von MiLe gilt $d = b * m$, wobei b die für MiLe gewählte Bitlänge und m die gewählte Minimallänge ist. P_{match} sei die Wahrscheinlichkeit, dass zwei Sequenzen als übereinstimmend klassifiziert werden. Für alle verwandten Ansätze bis auf

DCT-Fingerprinting gilt, dass P_{match} nur zwei Werte annehmen kann: Den durch Formel 6.6 gegebenen Wert bei Nichtübereinstimmung mindestens eines Tokens, beziehungsweise 1, wenn alle Tokens übereinstimmen. Wenn alle Tokens der beiden Sequenzen bis auf einen übereinstimmen, erhöht das nicht die Wahrscheinlichkeit dafür, dass die beiden Sequenzen als übereinstimmend klassifiziert werden. Dies ist ein Problem fast aller beschriebenen verwandten Ansätze: Sie sind wenig robust gegenüber Änderungen einzelner Tokens. Beim DCT-Fingerprinting-Ansatz wird dieses Problem adressiert. Der Ansatz gewinnt dadurch an Robustheit, dass mit einer gewissen Wahrscheinlichkeit zwei Sequenzen, die sich in einem einzelnen Token unterscheiden, dennoch denselben Hashwert zugewiesen bekommen. Die Autoren quantifizieren diese Wahrscheinlichkeit nicht [SC08]. Die Evaluationsergebnisse zeigen jedoch, dass dadurch ein Gewinn an Qualität gegenüber dem zugrundeliegenden Hash-Breaking-Algorithmus erzielt werden kann. MiLe besitzt diese Robustheits-Eigenschaft ebenfalls. Sie lässt sich in Abhängigkeit der in zwei Sequenzen übereinstimmenden Tokens quantifizieren. Sei b die für MiLe gewählte Bitlänge, m die Minimallänge für eine Übereinstimmung und q die innerhalb zweier Sequenzen in der entsprechenden Reihenfolge übereinstimmenden Tokens, dann gilt für die Wahrscheinlichkeit P_{match}, dass die zwei Sequenzen als übereinstimmend klassifiziert werden:

$$P_{match} = \frac{1}{2^{b(m-q)}} \quad 0 \leq q \leq m \tag{6.7}$$

Wenn q Null ist, dann entspricht die Wahrscheinlichkeit der oben berechneten unerwünschten Kollisionswahrscheinlichkeit. Je weiter sich q der Minimallänge m annähert desto größer wird die Wahrscheinlichkeit, dass die beiden Sequenzen als übereinstimmend erkannt werden. Wenn also zwei Sequenzen der Minimallänge in weiten Teilen übereinstimmen, dann ist die Wahrscheinlichkeit dafür, dass sie als übereinstimmend gewertet werden hoch im Vergleich zu anderen Ansätzen. Der Haupteinflussfaktor ist hierbei die Bitlänge b. Je kleiner b ist, desto höher ist die Wahrscheinlichkeit dafür, dass zwei zum Teil übereinstimmende Sequenzen als vollwertiger Match eingeordnet werden.

Eine weitere Möglichkeit der Parametrisierung ergibt sich - wie bei ShingleCloud - durch Einführung eines *Toleranzparameters*, der eine bestimmte Anzahl von Nichtübereinstimmungen zwischen mehreren übereinstimmenden Sequenzen ignoriert. Dieser wirkt sich jedoch nur auf Übereinstimmungen aus, die auch ohne den Toleranzparameter schon eine Länge der gegebenen Minimallänge m aufweisen. Er bewirkt, dass ein längerer, durch eine Nichtübereinstimmung unterbrochener Match als komplett angesehen wird. Da der Fingerprint bei der Berechnung der Ähnlichkeit mit einem Fenster mit einer Fenstergröße gleich der gewählten Minimallänge in Shingles zerlegt wird, sollte der Toleranzparameter bei Verwendung mindestens die Länge m aufweisen, da bei Nichtübereinstimmung der zu vergleichenden Fingerprints in einem Zeichen m aufeinanderfolgende Fenster nicht übereinstimmen.

Speicherbedarf

Eine wichtige Eigenschaft einer Fingerprinting-Technik im Rahmen dieser Arbeit ist der Bedarf an Speicherplatz, den der Fingerprint eines Dokument einnimmt. Je nach gewählter Bitlänger der Hashwerte kann die Menge der von K-Gram erzeugten Fingerprints einen höheren Speicherbedarf haben als das

zu Grunde liegende Dokument selbst. Da der Fingerprint jedoch als Metadatum des Dokuments behandelt werden soll ist es wünschenswert, dass er einen möglichst geringen Speicherbedarf aufweist. Der Speicherbedarf des MiLe-Fingerprints hängt lediglich von der Länge des Dokuments sowie der gewählten Bitlänge ab. Bei einem Dokument mit n Tokens und einer Mile-Bitlänge b benötigt der Fingerprint den folgenden Speicher:

$$L_{MiLe} = \frac{n * b}{8} Byte \tag{6.8}$$

Dadurch, dass die Bitlänge b normalerweise sehr klein ist, erreicht MiLe typischerweise einen Kompressionsfaktor gegenüber K-Gram von ungefähr 4-16 abhängig von der Bitlänge des für K-Gram gewählten Hashwertes. Eine höhere Kompression hat dabei auch oftmals eine geminderte Qualität zur Folge (siehe Abschnitt 6.3).

Optimierung des Laufzeitverhaltens

Ein weiterer Faktor für die Qualität eines Fingerprinting-Ansatzes ist gegeben durch dessen Laufzeitverhalten. Für den vorliegenden Anwendungsfall, der Validierung bestehender Variantenrelationen, ist das Laufzeitverhalten zwar nicht von großer Bedeutung, es gibt jedoch viele weitere Anwendungsfälle, wo eine gute Performanz wichtig ist (siehe 6.4.2). Daher wird im Folgenden das Laufzeitverhalten von MiLe analysiert und eine Optimierung für eine Anwendung auf großen Datenmengen vorgestellt.

Die Fingerprint-Länge eines MiLe Fingerprints ist direkt proportional zur Länge der zu Grunde liegenden Dokumente. Um zwei bestehende Fingerprints zu vergleichen, sind prinzipiell zwei Schritte notwendig:

- Erstellen einer Nachschlagetablle, beispielsweise in Form einer Hashtable, für alle n-Gramme aus Fingerprint 1
- Nachschlagen jedes n-Gramms aus Fingerprint 2 in der in Schritt 1 erstellten Tabelle

Für den Vergleich seien zwei Fingerprints bestehend aus jeweils i und j n-Grammen gegeben. Durch die direkte Proportionalität der Fingerprints zur Länge des zu Grunde liegenden Dokuments haben die Dokumente zu den beiden Fingerprints auch ungefähr die Längen i und j. Angenommen, dass i Schritte zur Erstellung der Tabelle und j Schritte zum Nachschlagen sowie jeweils ein Schritt für zusätzliche Berechnungen notwendig sind, hat MiLe eine Komplexität von circa $O(i + 2j)$. Wenn die Ähnlichkeit der Dokumente Null ist, dann strebt die Komplexität gegen $O(i + j)$ als untere Schranke für MiLe in der beschriebenen Form. Dies ist vergleichbar mit dem Laufzeitverhalten von K-Gram, wenn die Fingerprints nicht schon in einer Nachschlagetabelle gespeichert sind.

Nichtsdestotrotz kann die Performanz von K-Gram - und demnach allen darauf basierenden Ansätzen - erheblich gesteigert werden, wenn ein inverser Index (siehe [BYRN99]) zum Einsatz kommt. In einem solchen Index werden die n-Gramme als Schlüssel gespeichert, der auf alle Dokumente, die diesen Schlüssel enthalten verweist. Für eine Multiset-Berechnung wird zusätzlich die Anzahl erhoben mit der ein Shingle in einem Dokument auftritt. In Einsatzszenarien mit großen Mengen an Dokumenten, wie

der Websuche oder des Vergleichs von Dokumenten auf großen Korpora, ist ein solcher Index unerlässlich, da er eine sehr schnelle Berechnung der Abdeckungswerte eines Dokuments zu sehr vielen anderen Dokumenten ermöglicht. Das Laufzeitverhalten von MiLe kann ebenso durch einen solchen inversen Index verbessert werden. In einem solchen Fall verliert der Ansatz jedoch Flexibilität, da als Schlüssel des inversen Indexes für einen MiLe-Fingerprint die aus der Kombination von Bitlänge und Minimallänge bestehende Vergleichseinheit gewählt werden muss. Die Minimallänge muss also, wie auch die Bitlänge vor der Erzeugung des Indexes festgelegt werden und ist nicht, wie in einem Szenario, in dem kein Index zum Einsatz kommt, im Nachhinein flexibel anpassbar. Die Länge der Vergleichseinheit ist hierbei abhängig von der Größe des zu Grunde liegenden Korpus und der durchschnittlichen Länge der Dokumente zu wählen. Typischerweise ist die Vergleichseinheit etwa so groß wie die Bitlänge des Hashwertes, der für ein Verfahren wie K-Gram in diesem Fall verwendet werden würde.

Je größer der Korpus ist, desto mehr Speicherplatz benötigt ein invertierter Index. In Szenarien wie der Websuche kann ein solcher Index sehr schnell mehrere Terabyte groß werden [HBCH09]. Deshalb zielen Ansätze wie Winnowing, 0-mod-p oder Hailstorm darauf ab, die Anzahl der gewählten Shingles, die in den Index aufgenommen werden, zu verringern (siehe oben). Auch ein über einem MiLe-Fingerprint aufgebauter inverser Index kann komprimiert werden, ohne dabei die positiven Effekte des Fingerprints gegenüber den anderen Ansätzen, also die Robustheit gegenüber kleinen Änderungen, sowie die Berücksichtigung des Textzusammenhangs, aufzugeben. Hierzu wird ein weiterer Parameter - der Auslassfaktor o - eingeführt. Dieser gibt an, wieviele der sich aus den Shingles des Fingerprints ergebende Vergleichseinheiten bei der Konstruktion des Indexes ausgelassen werden sollen. Bei $o = 1$ wird jedes Shingle in den Index aufgenommen, bei $o = 2$ jedes zweite für $o = 3$ jedes Dritte und so weiter. Somit ergibt sich die folgende Kompression der Größe des inversen Indexes im Vergleich zu K-Gram:

$$\left|I_{MiLe}\right| = \frac{\left|I_{K-Gram}\right|}{o} \tag{6.9}$$

Um durch die Verwendung des inversen Indexes weiterhin den Textzusammenhang berücksichtigen zu können, werden die Vergleichseinheiten bei der Berechnung der Abdeckung entsprechend des Auslassfaktors gewichtet. Die Berechnung erfolgt dabei wie folgt:

1. Der MiLe-Fingerprint des Eingangsdokuments, das gegen den inversen Index verglichen werden soll, wird erzeugt und ein Zähler c initialisiert.

2. Für jedes n-Gramm dieses Fingerprints (also jede Vergleichseinheit) erfolgt ein Look-up im inversen Index, und

   a) wenn das n-Gramm nicht vorhanden ist, wird mit dem nächsten n-Gramm des Fingerprints fortgefahren.

   b) wenn das n-Gramm im Index vorhanden ist und für das verwiesene Dokument das n-Gramm zuvor nicht vorhanden war, wird der Betrag $m + o/w$ zu c addiert. Mit w wird parametrisiert, welcher Anteil der übersprungenen Shingles als übereinstimmend gewertet wird (siehe unten).

c) war das letzte n-Gramm im Index vorhanden und dieses n-Gramm ist es auch, so wird o zum Zähler addiert.

3. Zur Berechnung der Abdeckung erfolgt schließlich wiederum die Division von c durch die Länge der jeweiligen Fingerprints.

Die Länge der Fingerprints der Dokumente des Korpus sind im inversen Index nicht explizit enthalten. Somit muss diese zur Berechnung der entsprechenden Abdeckung noch zusätzlich vorgehalten werden. Da dies jedoch auch für alle anderen Ansätze gilt, kann dieser Umstand im Weiteren vernachlässigt werden. Die Idee hinter dem Ansatz ist, dass wenn nach dem Filtern der übersprungenen Shingles zwei aufeinanderfolgende Shingles für ein verwiesenes Dokument eine Übereinstimmung ergeben, dies auch für die übersprungenen Shingles angenommen werden kann. Somit erfolgt in diesem Fall eine Inkrementierung der gemeinsamen Sequenzen um o. Für den Fall, dass ein Shingle für ein Dokument den ersten Match darstellt, kann über die $o - 1$ übersprungenen vorhergehenden Shingles keine gesicherte Annahme getroffen werden. Daher wird der Parameter w eingeführt, der angibt, wie viele der übersprungenen Shingles als übereinstimmend gewertet werden. Ein sinnvoller Wert für w ist $0,5$.

Ein Nachteil der Verwendung eines inversen Index mit MiLe ist die Tatsache, dass aus dem Index der Fingerprint der Dokumente aus dem Korpus nicht reproduziert werden kann, daher kann in einem solchen Fall beispielsweise auch nicht der beschriebene Toleranzparameter verwendet werden. Deshalb ist es in verschiedenen Fällen sinnvoll, den Fingerprint der Dokumente zusätzlich zum Index zu speichern. Da sich ein MiLe-Fingerprint durch einen geringen Speicherverbrauch auszeichnet und der Speichergewinn, den der Auslassfaktors erbringt, höher ist, als die durch ein Speichern der Fingerprints verursachten Kosten, hat dies keine großen Nachteile zur Folge. Außerdem muss in diesem Fall die Länge der Fingerprints nicht zusätzlich gespeichert werden, da sie aus den gespeicherten Fingerprints berechnet werden kann.

6.2.2 MiLe für objektbasierte Dokumente

Eine Anwendung von MiLe auf andere Dokumenttypen soll im Folgenden anhand von PowerPoint-Dokumenten beschrieben werden. Die Erweiterung lässt sich jedoch auf viele objektbasierte Dokumenttypen übertragen. Ein Voraussetzung hierfür ist, dass die Objekte eines Dokuments eine gegebene Ordnung haben, beziehungsweise sich in eine solche Ordnung überführen lassen, die im Fall einer Wiederverwendung erhalten bleibt. Bei Text ist diese Voraussetzung durch die Ordnung der Worte innerhalb des Textes implizit gegeben. Im Fall von PowerPoint ist eine solche Ordnung durch die IDs der Objekte (sogenannter "Shapes") auf einer Folie gegeben. Eine weitere Herausforderung bei der Übertragung des MiLe-Algorithmus auf objektbasierte Dokumente ist die Gewichtung der Objekte. Bei Texten kann für den gegebenen Anwendungsfall davon ausgegangen werden, dass jedem Token (also jedem Wort) ungefähr dieselbe Bedeutung beizumessen ist, so dass jeder dasselbe Gewicht bei der Bildung des Fingerprints erhält. Mechanismen wie Stoppwortfilterung werden hier eingesetzt, um die Gewichtung bedeutungsloser Tokens zu verringern. Bei objektbasierten Dokumenten ist diese Voraussetzung nicht implizit erfüllt. Ein solches Dokument kann aus sehr vielen unterschiedlichen Objekten mit unterschiedlicher Bedeutung bestehen. So kann ein Diagramm in PowerPoint aus vielen unterschiedlichen Objekten zusammengesetzt sein. Es ist jedoch auch möglich, dasselbe Diagramm als einzelnes Objekt, beispielsweise in Form eines

Bildes, einzufügen. Der MiLe-Algorithmus muss hier gewährleisten, dass das aus vielen Objekten zusammengesetzte Objekt und das inhaltlich gleiche Bild im Fingerprint ungefähr dasselbe Gewicht erhalten. Der resultierende Fingerprint soll eine möglichst gute Repräsentation der Eigenschaften des Dokuments, bei möglichst kleinem Speicherverbrauch gewährleisten.

Auf Folien oder in Objekten enthaltener Text wird behandelt wie in der textbasierten Version von MiLe, das heißt jedes Wort geht in den Fingerprint ein. Bevor ein MiLe-Fingerprint für ein PowerPoint-Dokument erzeugt werden kann, sind Vorverarbeitungsschritte notwendig. Text wird hierbei auf dieselbe Art und Weise wie bei der textbasierten Version von MiLe vorverarbeitet. Objekte werden folienweise anhand ihrer ID sortiert und auf die kleinstmögliche Granularitätsstufe gebracht, da es in vielen Applikationen möglich ist, Elemente zu gruppieren und zu Elementen auf einem höheren Aggregationsniveau zusammenzufassen. Durch eine Zerlegung eines gruppierten Objekts ist gewährleistet, dass dieses dieselbe Gewichtung und denselben Fingerprint erhält wie in nicht gruppierter Form.

Für eine Gewichtung der Objekte in einem PowerPoint-Dokument gibt es folgende Möglichkeiten:

- **Naiver Ansatz**: Jedes Objekt erhält dasselbe Gewicht, ähnlich wie bei Worten, unabhängig von Größe, Typ oder Inhalt.

- **Gewichtung anhand der Folienfläche**: Ein Objekt wird anhand seiner Fläche auf einer Folie gewichtet. Ein Objekt mit größerer Fläche erhält dabei mehr Gewicht als ein Objekt mit geringerer. Das setzt voraus, dass ein größeres Objekt mehr Bedeutung hat als ein kleineres. In bestimmten Fällen, wie z.B. bei Bildern ist diese Voraussetzung plausibel. Bei einfachen Objekten, wie zum Beispiel geometrischen Formen, ist die Größe weniger aussagekräftig. Bei einer solchen Gewichtung muss eine Bezugsgröße gewählt werden, anhand der die Gewichtung erfolgt. Dies ist hier die auf einer Folie vorhandene Fläche sein. Zusätzlich kann die Gewichtung noch auf eine maximale Anzahl von Teilsequenzen pro Objekt oder pro Folie im Fingerprint normalisiert werden.

- **Gewichtung anhand der Gesamtfläche der Objekte**: Die Gewichtung erfolgt wie bei Punkt 2 anhand der Fläche eines Objekts. In diesem Fall dient als Bezugsgröße jedoch die Gesamtfläche aller Objekte auf einer Folie, die aufgrund von überlappenden Objekten auch größer sein kann als die Gesamtfläche der Folie. Auch hier kann die Gewichtung noch auf eine maximale Anzahl von Teilsequenzen pro Objekt oder pro Folie im Fingerprint normalisiert werden.

- **Gewichtung anhand des Objekttyps**: Verschiedene Objekttypen können eine unterschiedliche inhaltliche Bedeutung für ein Dokument haben. So vermittelt im Allgemeinen ein Bild mehr Informationen als eine einfache geometrische Form und kann somit stärker gewichtet werden. Selbiges gilt für ein Video gegenüber einem Bild.

- **Kombinierte Gewichtung**: Hier werden sowohl der Typ als auch die Größe eines Objekts berücksichtigt. Einfache geometrische Formen werden zum Beispiel einfach gewichtet, während Bilder anhand ihrer Größe gewichtet werden.

- **Gewichtung anhand des Inhalts**: Die vermutlich beste Gewichtung eines Objektes lässt sich anhand seines Inhalts vornehmen; da dieser jedoch nur sehr schwer automatisch zu erfassen oder zu vergleichen ist, wird diese Möglichkeit im Rahmen dieser Arbeit nicht weiter verfolgt.

Die Hashwerte der einzelnen Objekte selbst können wie im Fall von Tokens aus Text in Bitfolgen umgewandelt werden, zum Beispiel mit Hilfe von Hash-Techniken wie MD5. Anstatt des Objektes selbst ist es auch möglich nur bestimmte Eigenschaften, wie zum Beispiel die Größe oder Farbe des Objekts für die Berechnung des Hashwerts zu verwenden. Die Umsetzung der Gewichtung erfolgt durch entsprechende Replikation des Hashwerts im resultierenden Fingerprint.

Die Gewichtungsmethoden wurden folgendermaßen umgesetzt: Der naive oder **einfache Ansatz** ("Einfach") wurde wie beschrieben umgesetzt. Alternativ dazu wurden einige Gewichtungsmethoden eingesetzt, wobei jeweils ein Grenzwert für die Anzahl vergebenen Fingerprints pro Folie festgelegt wurde (z.B. 50). Bei der Gewichtung in Abhängigkeit der Fläche eines Objekts gibt es zwei Möglichkeiten: Bei der *Gewichtung anhand der Foliengröße* ("Folienfläche"), erhält ein einzelnes Objekt den Anteil der zu vergebenden Fingerprints, die seinem Anteil an der Gesamtfläche der Folie entspricht. Dies kann jedoch dazu führen, dass mehr Fingerprints vergeben werden als der Grenzwert vorgibt. Dies geschieht, wenn die Gesamtfläche der Shapes auf einer Folie die der Folie selbst übersteigt. Bei der **Gewichtung anhand der Gesamtfläche** ("Gesamtfläche") der Shapes einer Folie geschieht dies nicht. Ein Shape wird seinem prozentualen Anteil an der Gesamtfläche der Shapes pro Folie entsprechend gewichtet. Alternativ dazu kann man auch **beide Ansätze kombinieren** ("Gemischt"), indem man den ersten Ansatz wählt, wenn die Gesamtfläche der Shapes einer Folie kleiner ist als die der Folie und letzteren im anderen Fall. Eine Gewichtung anhand des Typs eines Shapes wird in den beschriebenen Fällen implizit umgesetzt. Es werden lediglich Shapes überhaupt stärker als Text gewichtet, wenn es sich um inhaltlich komplexe Objekte handeln kann, wie es zum Beispiel bei Bildern oder Diagrammen der Fall ist. Basis-Formen wie Kreise oder Rechtecke, werden weiterhin einfach gewichtet.

Ein objektbasiertes Dokument kann je nach Typ und Anzahl der enthaltenen Objekte vergleichsweise lange Fingerprints erzeugen. Entsprechend groß ist jedoch auch die Anzahl der Fingerprints, die andere Ansätze wie K-Gram oder Winnowing aus einem solchen Dokument erzeugen.

6.3 Evaluation von MiLe

Da Korpora für die Evaluation von Fingerprinting-Techniken fast nur in Textform existieren, fällt die Evaluation der textbasierte Version von MiLe ausführlicher aus als die der objektbasierten Version. Für letztere kann lediglich der aus der Evaluation des LIS.KOM-Frameworks extrahierte Korpus verwendet werden, während die textbasierte Version von MiLe auf allen genannten Korpora getestet wird. Im Folgenden werden die Ergebnisse der Test zusammen mit dem genauen Setting der jeweiligen Auswertung auf den unterschiedlichen Korpora beschrieben.

6.3.1 Annotierter TREC-Korpus

Auf dem annotierten TREC-Korpus wurden MiLe, K-Gram, 0-mod-p und Winnowing getestet und verglichen. Auf eine Implementierung der übrigen Algorithmen wurde verzichtet. Wenn MiLe die Qualität von K-Gram erreicht oder übertrifft, ist dies auch nicht notwendig, da keiner der übrigen Ansätze bessere Ergebnisse liefert als K-Gram [SC08]. Die Ergebnisse von 0-mod-p und Winnowing dienen dazu, die Ergebnisse von MiLe mit denen selektiver Algorithmen vergleichbar zu machen. 0-mod-p komprimiert für

große p sehr stark somit kann der Speicherverbrauch von MiLe bei vergleichbaren Ergebnissen beurteilt werden. Als Vorverarbeitungsschritte wurden Sonderzeichen und Satzzeichen entfernt und alle Tokens klein geschrieben. Wie bei der Auswertung des ShingleCloud-Ansatzes auf diesem Korpus wird auch hier

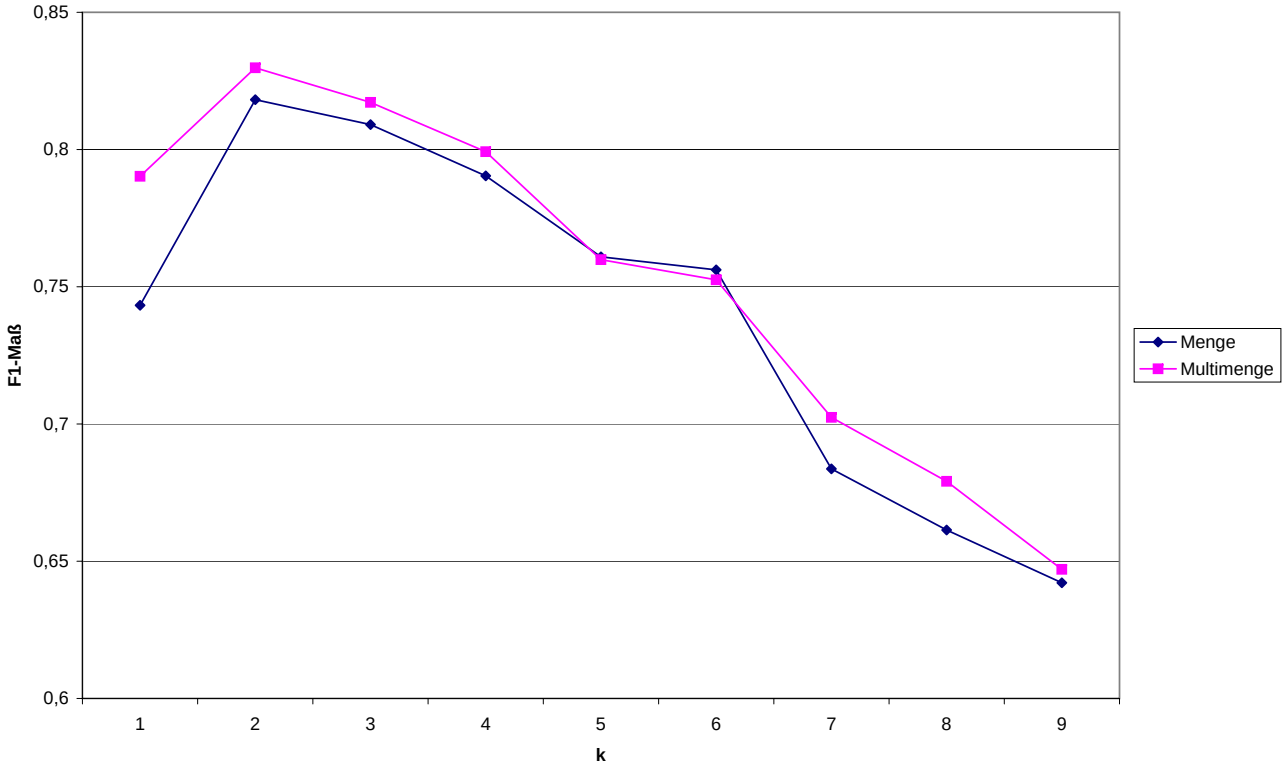

Abbildung 6.2.: Ergebnisse von K-Gram auf dem annotierten TREC-Korpus

eine zehnfache Kreuzvalidierung angewendet. Die Ergebnisse werden wiederum über 100 Durchläufe gemittelt, um eine hohe Konfidenz zu erreichen (siehe 5.5.1). Anhand der Ergebnisse der getesteten Algorithmen lassen sich die Eigenschaften des jeweiligen Algorithmus gut darstellen. Abbildung 6.2 zeigt die Qualität von K-Gram auf dem TREC-Korpus. Wie zu sehen ist, erreicht der Ansatz für eine Fenstergröße von $k = 2$ die besten Ergebnisse. Eine Fenstergröße von eins ist zu klein und gewichtet damit häufige Worte zu stark. Ab einer Fenstergröße von vier wird der Algorithmus zu streng und viele Übereinstimmungen werden nicht erkannt. Es zeigt sich weiterhin, dass eine Betrachtung der Fingerprints als Multimenge (siehe 5.1.3), fast durchweg etwas bessere Ergebnisse liefert als die in der Literatur beschriebene mengenbasierte Berechnung der Abdeckung. Dadurch kommt es jedoch zu Performance-Einbußen, da zusätzlicher Rechenaufwand im Vergleich zum originalen Algorithmus notwendig ist. Die Ergebnisse von 0-mod-p entsprechen den Erwartungen. Je größer der Parameter p gewählt wird, desto stärker komprimiert der Algorithmus und desto schwächer werden die Ergebnisse. Abbildung 6.3 zeigt die Auswertung des Ansatzes für $k = 2, 3$, da diese Konfiguration die besten Ergebnisse liefert. Die Schwankungen nach oben und unten zeigen die Anfälligkeit des Algorithmus. Wenn für die Erkennung von Wiederverwendung relevante Shingles durch den Ansatz ausgefiltert wurden, wirkt sich das negativ auf die Qualität aus. Andererseits, kann ein bestimmter Wert von p auch dazu führen, dass eine "bessere Auswahl" an Shingles getroffen wird. Wenn häufig auftretende, nicht relevante Shingles durch den Ansatz ausgewählt werden, können die Ergebnisse dadurch ebenso verfälscht werden.

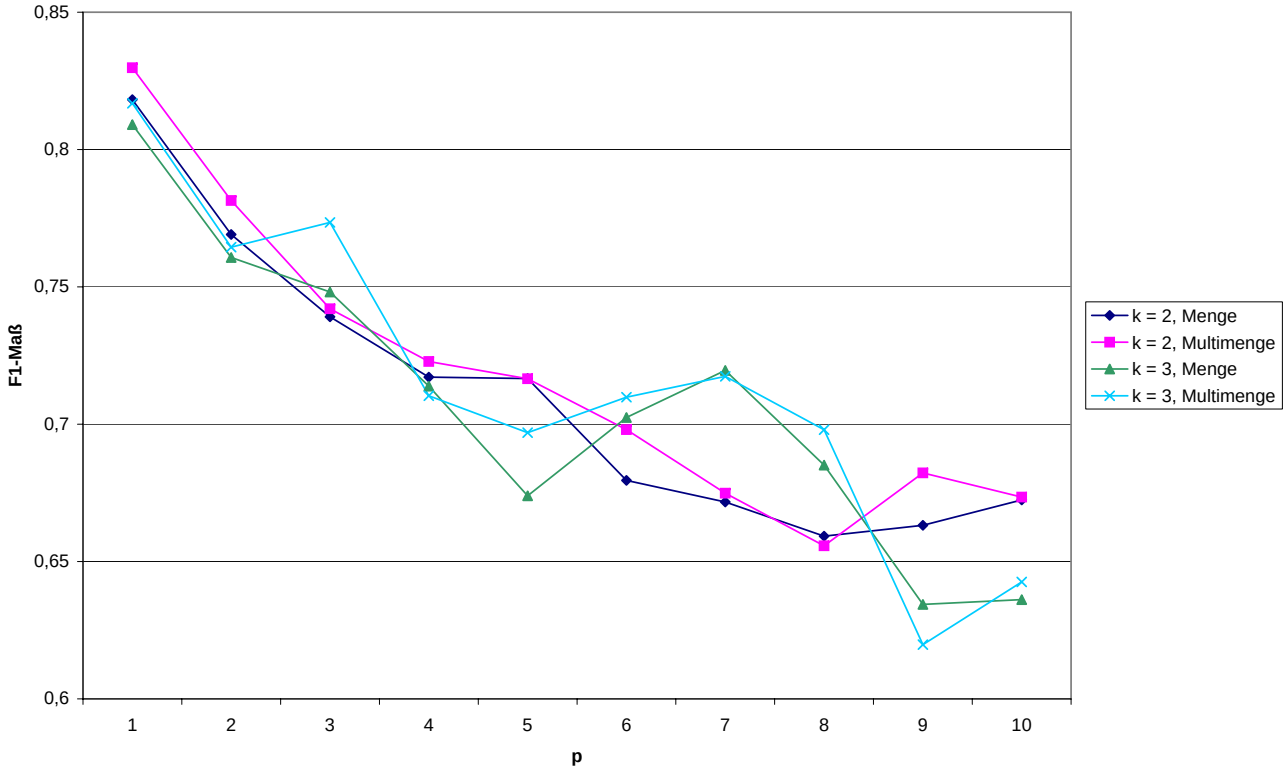

Abbildung 6.3.: Ergebnisse von 0-mod-p auf dem annotierten TREC-Korpus

Winnowing (Abbildung 6.4) zeigt ähnliches Verhalten wie 0-mod-p. Die Auswirkung der besseren Abdeckung von Übereinstimmungen der Mindestlänge w lässt sich jedoch gut anhand der gemilderten Schwankungen gegenüber 0-mod-p erkennen. Dadurch, dass sich der Parameter w bei Winnowing nur ungefähr halb so stark auf die Selektivität des Algorithmus auswirkt wie p bei 0-mod-p, muss w entsprechend groß gewählt werden um denselben Kompressionsgrad zu erreichen. Deshalb fällt die Qualität von Winnowing mit wachsendem w auch nicht so stark ab. Insgesamt sind die Ergebnisse jedoch qualitativ ähnlich denen eines bezüglich des Kompressionsgrades vergleichbaren 0-mod-p.

In Abbildung 6.5 ist die Auswertung von MiLe auf dem annotierten TREC-Korpus zu sehen. Es wurde die Qualität des Algorithmus für verschiedene Bitlängen in Abhängigkeit von der Minimallänge gemessen. Der Toleranzparameter ist in diesen Evaluationen null. Erwartungsgemäß führen größere Bitlängen insbesondere für kleine Minimallängen zu besseren Ergebnissen. Wenn die Minimallänge klein ist, ist bei kleinen Bitlängen die Kollisionswahrscheinlichkeit sehr hoch und die falsch positiven Befunde verfälschen das Resultat stark. Je höher die Bitlänge, desto geringer ist die Auswirkung dieses Effekts. Bei kleinen Bitlängen muss die Minimallänge entsprechend hoch gewählt werden, um annehmbare Resultate zu erzielen. Wie das Diagramm zeit, wirkt sich für höhere Minimallängen die Bitlänge kaum noch aus. Für eine Minimallänge von acht ergibt sich für alle Bitlängen ungefähr derselbe Wert. Da der Anwendungsfall für den MiLe-Fingerprint in dieser Arbeit keinen invertierten Index erfordert, wird auf das Verhalten von MiLe bei Anwendung des oben beschriebenen Auslassfaktors an dieser Stelle nicht weiter eingegangen. Die Qualität des Algorithmus ist jedoch auch mit Auslassfaktor deutlich höher als bei an-

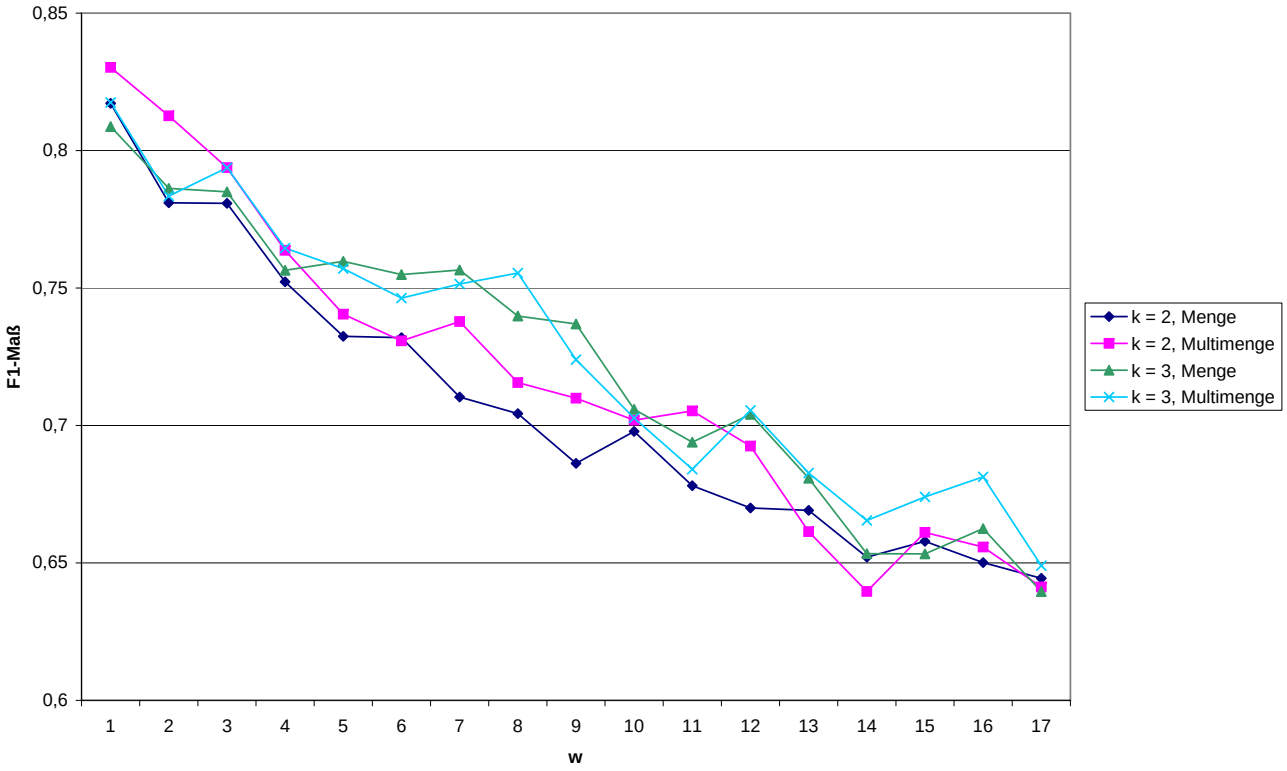

Abbildung 6.4.: Winnowing-Ergebnisse auf dem annotierten TREC-Korpus

deren, ähnlich stark komprimierenden Ansätzen. Eine ausführlichere Auswertung des Auslassfaktors auf dem annotierten TREC-Korpus findet sich in Anhang A.3 dieser Arbeit.

Das Diagramm in Abbildung 6.6 vergleicht eine Auswahl der Ergebnisse der verschiedenen Ansätze in unterschiedlichen Konfigurationen. Für MiLe ist dabei die Bitlänge gefolgt von der Minimallänge gegeben. Bei Anwendung des Auslassfaktors ist dieser zusätzlich in Klammern angegeben. MiLe4,5 (o=6) heißt also: Bitlänge vier, Minimallänge fünf und Auslassfaktor sechs. Es zeigt sich, dass MiLe in der Konfiguration mit Bitlänge fünf und Minimallänge vier signifikant bessere Ergebnisse liefert als die beste Konfiguration des K-Gram-Ansatzes. Der Speicherbedarf von MiLe beträgt dabei - abhängig von der gewählten K-Gram-Bitlänge - weniger als ein Drittel dessen von K-Gram. Auch in Konfigurationen mit vier Bit liefert MiLe vergleichbar gute Ergebnisse. Die Konfiguration mit drei Bit liefert zwar schwächere Ergebnisse als K-Gram, aber teilweise deutlich bessere Resultate als 0-mod-p oder Winnowing. Diese wurden jeweils in Konfigurationen getestet, in denen ungefähr 1/4 bzw. 1/6 der Fingerprints ausgewählt wurden. Der Speicherverbrauch ist dabei bei vergleichbaren Ergebnissen deutlich geringer: Ein 3-Bit-MiLe braucht $3n$ Bit für ein Dokument der Länge n. 0-mod-4 braucht bei Verwendung von 16 Bit Hashwerten $4n$ Bit bei deutlich schwächerer Performance. In einem Szenario, in dem ein invertierter Index benötigt wird, kann der Auslassfaktor eingesetzt werden. Auch hier zeigt sich, dass MiLe bei einem ähnlichen Kompressionsgrad - der Auslassfaktor o besitzt ungefähr die gleiche Kompressionsstärke wie p - deutlich bessere Ergebnisse aufweist. K-Gram lässt sich - wie gezeigt - durch Verwendung eines Multimengen-Ansatzes noch verbessern, während sich bei MiLe eine Verbesserung durch Verwendung des Toleranzparameters erwirken lässt. Beide wurden jedoch hier der Übersichtlichkeit halber weggelassen, da die Auswirkung auf das Gesamtbild der Evaluation eher gering sind und auch in diesem Fall MiLe signifikant bessere

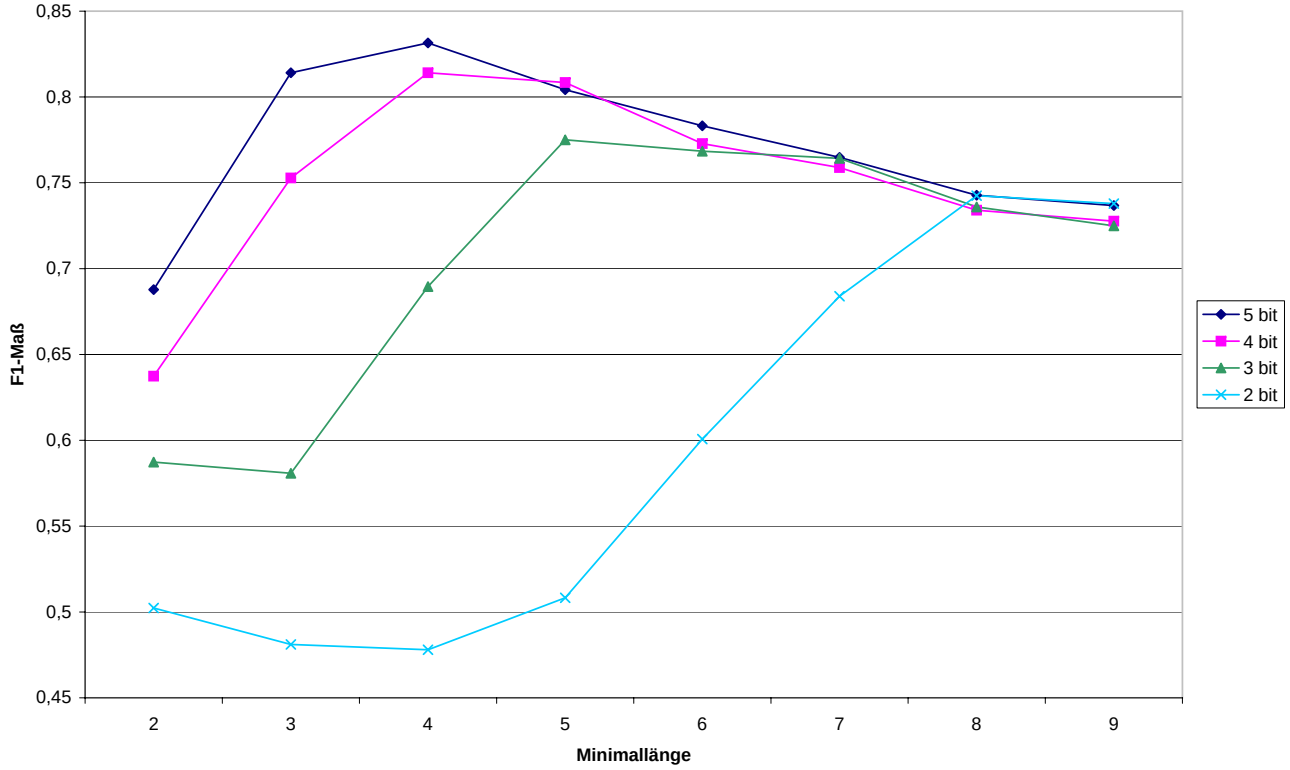

Abbildung 6.5.: MiLe-Ergebnisse auf dem annotierten TREC-Korpus

Ergebnisse liefert als K-Gram. Ein weiterer Vergleich der Ansätze unter Berücksichtigung der genannten Parameter findet sich in Anhang A.3.

Auf dem annotierten TREC-Korpus ist MiLe folglich in der Lage, bessere Ergebnisse als K-Gram und somit auch alle anderen Fingerprinting-Ansätze zu liefern und gewährleistet dabei einen deutlich geringeren Speicherverbrauch.

6.3.2 METER-Korpus

Genau wie beim annotierten TREC-Korpus und ebenso wie bei der Evaluation des ShingleCloud-Ansatzes auf diesem Korpus wurde auf dem METER-Korpus eine stratifizierte Zehnfach-Kreuzvalidierung mit 100 Durchläufen durchgeführt. Auch hier wurde ein baumbasierter J48-Klassifikator verwendet. Als Feature wurde wie beschrieben die Abdeckung des Zeitungsartikels durch die gesammelten Presseagentur-Texte gewählt. Für die hier gezeigten Ergebnisse wurden aus den Originaltexten Sonderzeichen, Satzzeichen sowie Stoppwörter entfernt. Mit anderen Vorverarbeitungsschritten unterscheiden sich einzelne Ergebnisse; prinzipiell ändert sich am Ergebnis der Evaluation jedoch nichts. Ergebnisse für einen weiteren Vorverarbeitungsprozess sind im Anhang A.2 zu finden.

Tabelle 6.1 zeigt die Ergebnisse für K-Gram, 0-mod-p und MiLe in verschiedenen Konfigurationen. Es sind die Ergebnisse in den drei einzelnen Kategorien (WD - wholly derived, PD - partially derived und ND - non derived) sowie der über die Verteilung der Muster gewichtete Durchschnitt daraus als Gesamtergebnis aufgeführt. Die letzte Spalte gibt das Konfidenzintervall der ermittelten Ergebnisse an.

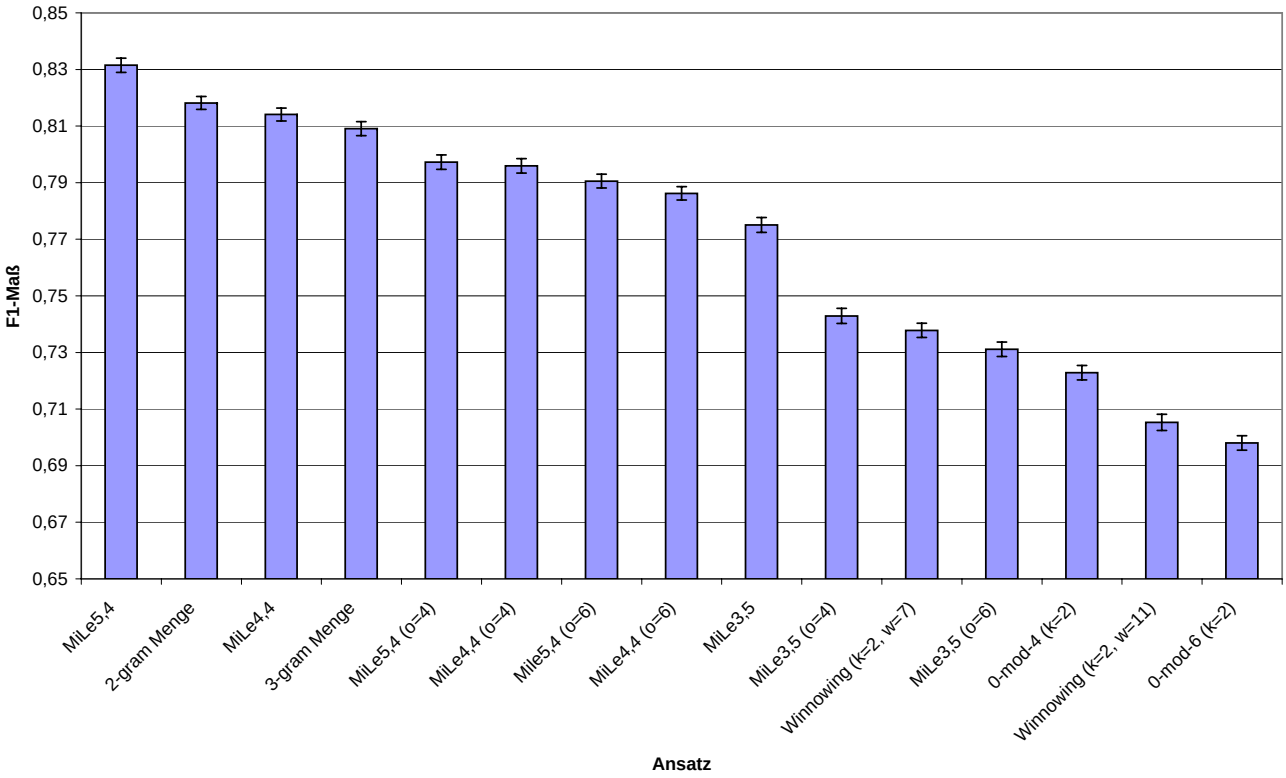

Abbildung 6.6.: Vergleich von K-Gram, Winnowing, 0-mod-p und MiLe auf dem annotierten TREC-Korpus

Wiederum ist zum Vergleich die Referenzgüte (Baseline) angegeben, die ein naiver Ansatz erzielt, wenn er alle Dokumentpaare der Kategorie mit den meisten Mustern zuordnet. Wie bei der Auswertung von ShingleCloud auf diesem Korpus in Kapitel 5.5.2 festgestellt, zeigt der 1-Gram-Ansatz, bedingt durch die Art und Weise wie die Annotation der Wiederverwendung im Korpus erfolgte, überproportional bessere Ergebnisse als alle anderen Ansätze.

Der Effekt ist hier in der Tat so groß, dass keiner der anderen Ansätze in die Nähe von 1-Gram kommt. Davon abgesehen liefert MiLe in einer Konfiguration mit sechs Bit und Minimallänge drei Ergebnisse, die besser als die Ergebnisse von 2- bzw. 3-Gram und alle Ergebnisse von 0-mod-p sind. Interessant ist weiterhin, dass 0-mod-4 mit $k = 2$ bessere Ergebnisse liefert als der zu Grunde liegende 2-Gram. Dies ist durch die Schwankungen, denen 0-mod-p unterlegen ist und die sich auch positiv auswirken können zu erklären. Auf diesem Korpus benötigt MiLe offensichtlich größere Bitlängen um konkurrenzfähige Ergebnisse zu liefern. Der Grund hierfür ist die Verwendung einer Stoppwortliste. Durch Filterung häufiger Wörter wird generell die Länge der übereinstimmenden Sequenzen verkürzt. Eine große Minimallänge wirkt sich daher negativ auf die Qualität aus, da viele Übereinstimmungen übersehen werden, daher muss die Minimallänge entsprechend klein gewählt werden. Um ungewollte Kollisionen, die sich wiederum negativ auswirken, zu vermeiden, muss die Bitlänge entsprechend vergrößert werden.

Mit anderen Vorverarbeitungstechniken ist es jedoch auch möglich mit einer Bitlänge von vier zufriedenstellende Werte zu erhalten. Und auch hier lässt sich mit MiLe auf dem METER-Korpus stets das zweitbeste Ergebnis hinter 1-Gram erzielen.

Tabelle 6.1.: Vergleich der Fingerprinting-Ansätze auf dem METER-Korpus

Ansatz	Wholly-Derived	Partially-Derived	Non-Derived	Gewichtetes Mittel	Konfidenz-intervall
1-gram	0,728	0,639	0,613	**0,664**	± 0,0002
2-gram	0,682	0,609	0,507	**0,612**	± 0,0007
3-gram	0,667	0,608	0,397	**0,582**	± 0,0003
0 mod 4 (k = 3)	0,625	0,619	0,310	**0,554**	± 0,0005
0 mod 4 (k = 2)	0,666	0,627	0,542	**0,622**	± 0,0008
0 mod 6 (k = 3)	0,583	0,585	0,167	**0,494**	± 0,0007
0 mod 6 (k = 2)	0,660	0,573	0,335	**0,551**	± 0,0006
Winnowing (k = 2, w = 7)	0,598	0,610	0,389	**0,558**	± 0,0007
Winnowing (k = 3, w = 7)	0,628	0,585	0,299	**0,538**	± 0,0006
Winnowing (k = 2, w = 11)	0,555	0,615	0,427	**0,554**	± 0,0013
Winnowing (k = 3, w = 11)	0,570	0,583	0,115	**0,478**	± 0,0007
MiLe5,4 (5 bit, mml 4)	0,631	0,564	0,506	**0,574**	± 0,0010
MiLe6,3 (6 bit, mml 3)	0,690	0,636	0,537	**0,633**	± 0,0007
MiLe6,3 (o = 4) (6 bit, mml 3)	0,612	0,610	0,507	**0,589**	± 0,0004
MiLe6,3 (o = 6) (6 bit, mml 3)	0,582	0,571	0,429	**0,544**	± 0,0007
Baseline	0,3402	0,4442	0,2155	**0,4442**	-

6.3.3 20-Newsgroups-Korpus

Der 20-Newsgroups-Korpus besitzt keine Annotationen oder Kategorien. Daher wurde MiLe auf diesem Korpus evaluiert, indem dessen Ergebnisse mit den Ergebnissen des besten vergleichbaren Ansatzes, also K-Gram, verglichen und die Abweichung im Vergleich zu 0-mod-p bestimmt wurde. Die Auswertungen auf den anderen beiden Korpora haben gezeigt, dass MiLe meistens etwas bessere Ergebnisse liefert als K-Gram, weshalb eine Abweichung nicht notwendigerweise bedeuten muss, dass MiLe die Ähnlichkeit der Dokumente falsch beurteilt und K-Gram richtig. Es wurden lediglich Sonderzeichen aus den Originaltexten der Newsgroups entfernt, ansonsten wurden keine Vorverarbeitungschritte durchgeführt. Es wurden 32 Bit Hashwerte für K-Gram, benutzt, da bei geringere Bitlängen (z.B. 16 Bit) deutliche, durch Kollisionen bedingte Abweichungen auftraten. Vergleiche von Stichproben der unterschiedlichen Konfigurationen des K-Gram-Ansatzes ergaben, dass auf diesem Korpus eine Fenstergröße von drei die verlässlichsten Ergebnisse liefert. Dies entspricht auch den Erfahrungen aus der Literatur (z.B. [LMD01], [SC08]). Wiederverwendung in Newsgroups findet, wie auch zum Beispiel in Blogs, üblicherweise in Form von Zitaten statt. Die zitierten Beiträge werden dabei eher selten angepasst oder überarbeitet. So ist zu erwarten, dass Überlappungen in längere Sequenzen auftreten. Dies ist auch ein Grund dafür, warum 3-Gram hier besser funktioniert, obwohl 2-Gram bzw. 1-Gram auf den anderen Korpora bessere Ergebnisse aufweisen. Der Korpus ist ein typisches Anwendungsbeispiel für Information-Tracking (siehe 6.4).

Für die erste Auswertung wurden für alle Dokumentenpaare die Abdeckungsmaße ab einer Ähnlichkeit von 20 % verglichen und die absolute Abweichung zu 3-Gram berechnet. Zusätzlich wurde untersucht wieviel Prozent der ermittelten Ähnlichkeiten um mehr als 5, 10 und 20 % abweichen. Tabelle 6.2 zeigt die Ergebnisse dieser Auswertung. Es wurden die Abweichungen von MiLe mit 3, 4 und 5 Bit mit 0-mod-3 und 0-mod-4 verglichen. Je größer der Parameter p bei 0-mod-p ist, desto größer ist auch die Abweichung. Es zeigt sich, dass MiLe in allen Fällen eine deutlich geringere Abweichung zu 3-Gram aufweist als 0-mod-p. Mit höheren Bitlängen verringert sich die Abweichung von MiLe zu 3-Gram. Eine Ausnahme bildet hier die Abweichung im 5 %-Bereich, welche für 5 Bit im Vergleich zu 4 Bit wieder leicht ansteigt. Dies könnte ein Hinweis darauf sein, dass MiLe hier genauere Ergebnisse liefert als 3-Gram. Die Zahl in Klammern in der mittleren Spalte gibt die Gesamtzahl der Dokumentpaare an, die eine Abdeckung von mehr als 20 % aufweisen. Die Zahlen in Klammern in der letzten Spalte geben jeweils die absolute Anzahl der Dokumentpaare an, für die die ermittelten Ähnlichkeitswerte um den entsprechenden Wert abweichen.

6.3.4 LIS.KOM-Korpus

Die Evaluation von MiLe auf dem LIS.KOM-Korpus setzt sich aus zwei Teilen zusammen: Die textbasierte Version des Ansatzes wird auf den für Word-Dokumente erfassten Variantenrelationen evaluiert, während die objektbasierte Version auf den PowerPoint-Variantenrelationen getestet wird.

Der LIS.KOM-Korpus ergab sich aus der Evaluation des LIS.KOM-Frameworks. Mit Hilfe des MiLe-Fingerprints soll es möglich sein, Variantenrelationen zwischen text- sowie objektbasierten Wissensdokumenten zu validieren. Wie sich bei der Evaluation des LIS.KOM-Frameworks zeigt (siehe Kapitel 8),

Tabelle 6.2.: Abweichungen von MiLe zu 3-gram auf dem 20-Newsgroups-Korpus

Ansatz	**Anzahl relevanter Dokumentpaare**	**Absolute Abweichung**	**Abweichung pro Gruppe** (Anzahl abweichender Dokumentpaare)
$MiLe_{3,9,0}$ 3 bits, mml 9	66129	0,014	≥ 20 %: 0,026 (1731) ≥ 10 %: 0,034 (2277) ≥ 5 %: 0,064 (4261)
$MiLe_{4,7,0}$ 4 bits, mml 7	65324	0,011	≥ 20 %: 0,008 (515) ≥ 10 %: 0,015 (1011) ≥ 5 %: 0,042 (2723)
$MiLe_{5,6,0}$ 5 bits, mml 6	65682	0,010	≥ 20 %: 0,003 (210) ≥ 10 %: 0,012 (767) ≥ 5 %: 0,044 (2904)
0 mod 3 with k = 3	74745	0,036	≥ 20 %: 0,053 (3993) ≥ 10 %: 0,189 (14092) ≥ 5 %: 0,382 (28544)
0 mod 4 with k = 3	76561	0,047	≥ 20 %: 0,095 (7280) ≥ 10 %: 0,263 (20115) ≥ 5 %: 0,476 (36418)

werden Variantenrelationen im LIS-KOM-Framework mit sehr hoher Verlässlichkeit erfasst. Es kommt daher sehr selten vor, dass eine Variantenrelation zwischen zwei Dokumenten erfasst wird, die in keinerlei Zusammenhang stehen. Oftmals kommt es jedoch vor, dass Dokumente kopiert werden um als Vorlage zu dienen, da zum Beispiel die Layout-Vorlage des kopierten Dokuments übernommen werden soll, oder seien Struktur. Zwischen dem Dokument und seiner Kopie wird dabei eine Variantenrelation erzeugt. Für die meisten Anwendungsszenarien sind Relationen zwischen Dokumenten, die lediglich strukturelle, jedoch keine inhaltliche Verwandtschaft haben, uninteressant, beziehungsweise sogar als störend anzusehen. Ein Ziel der Validierung von Variantenrelationen ist es deshalb, inhaltlich relevante Relationen von den beschriebenen "Template"-Relationen zu unterscheiden. Der LIS.KOM-Korpus enthält daher alle, während der Evaluation des LIS.KOM-Frameworks für PowerPoint- und Word-Dokumente gesammelten Variantenrelationen. Diese wurden durch manuelle Bewertung in unterschiedliche Gültigkeitsstufen eingeteilt (siehe 8.3). Interessant für die Evaluation des MiLe-Fingerprints sind hierbei jedoch nur zwei Kategorien für eine solche Variantenrelation: *Gültig* und *Vorlage*. Die erste Kategorie umfasst alle als gültig bewerteten Variantenrelationen, während die letztere sowohl die als Template-Relationen eingestuften Relationen, als auch, soweit vorhanden, die als ungültig bewerteten beinhaltet. Für detailliertere Erläuterungen der Bewertungskriterien, -kategorien und Ergebnisse wird auf Kapitel 8 verwiesen.

Die manuelle Beurteilung der Variantenrelationen für Word-Dokumente ergab insgesamt 355 Relationen, von denen 325 als gültig und die übrigen 30 als Vorlage eingeordnet sind. Der MiLe-Fingerprint soll nun die gültigen von "Template"-Relationen unterscheiden. Durch den MiLe-Fingerprint wird für jedes durch eine Variantenrelation gegebenes Dokumentenpaar ein Abdeckungstupel berechnet. Mit Hilfe eines geeigneten Schwellwerts kann dann eine Relation als gültig oder nicht gültig kategorisiert werden. Dabei genügt es, wenn eins der beiden Abdeckungsmaße über dem Schwellwert liegt. Da die Zahl der als Vorlage eingestuften Relationen vergleichsweise gering ist, ergibt sich ein sehr hoher Referenzwert für das F1-Maß auf diesem Korpus. Er liegt in diesem Fall bei 91,5 %. Für die Evaluation wurde ein fünffacher Kreuzvalidierungsansatz gewählt. Dabei wurde für jedes der Word-Dokumente des Korpus jeweils ein MiLe-Fingerprint mit Bitlängen von zwei bis sechs Bit erstellt. Für die Auswertung wurden Minimallängen von eins bis zehn berücksichtigt. Das Diagramm in Abbildung 6.7 zeigt die Qualität des MiLe-Ansatzes auf dem textbasierten LIS.KOM-Korpus für verschiedene Bitlängen in Abhängigkeit von der gewählten Minimallänge. Die Referenzgüte (Baseline) ist zum Vergleich gegeben.

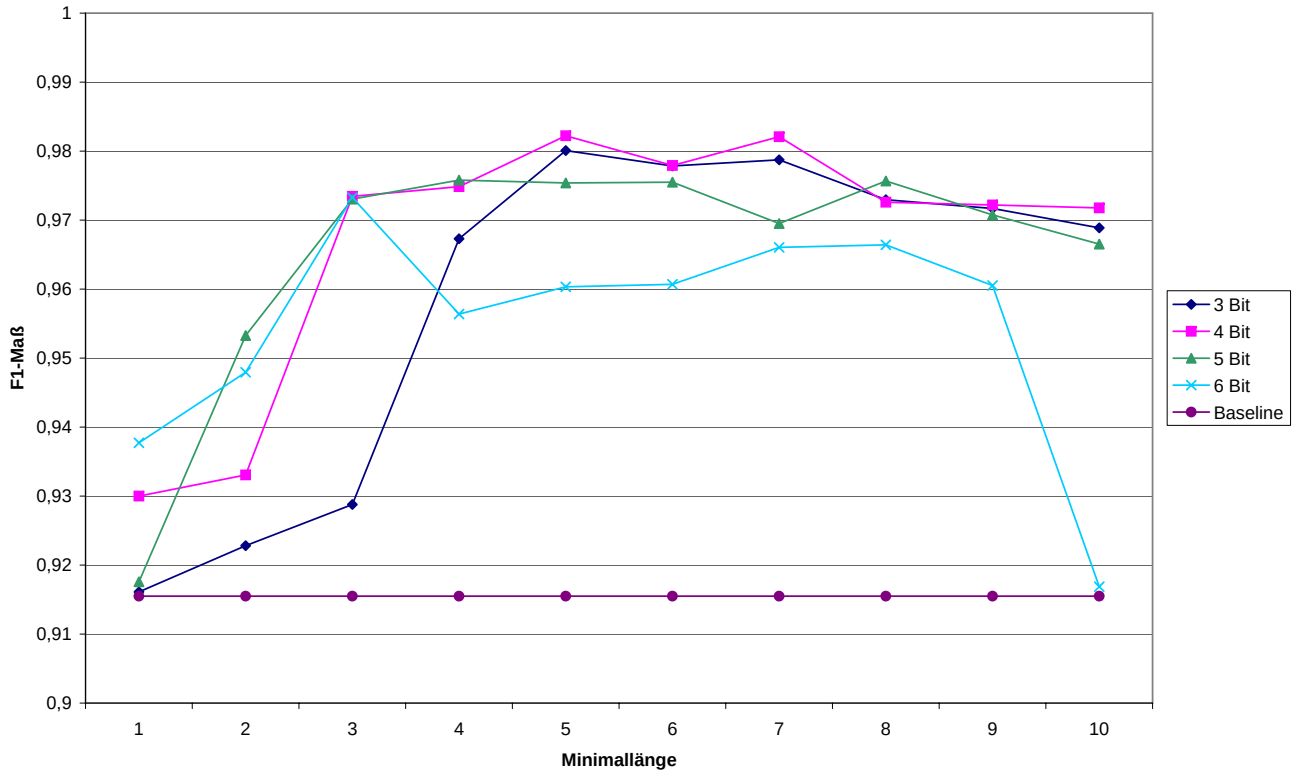

Abbildung 6.7.: Qualität des textbasierten MiLe-Fingerprint auf dem LIS.KOM-Korpus

Das Diagramm zeigt, dass MiLe in den meisten der gezeigten Parametrisierungen in der Lage ist, signifikant bessere Ergebnisse zu liefern als die Referenzgüte. Erwartungsgemäß braucht eine Konfiguration mit geringer Bitlänge (wie zum Beispiel drei) eine größere Minimallänge um gute Ergebnisse zu liefern. Ebenso nimmt die Qualität ab, wenn die Bitlänge zu groß gewählt wird und die Robustheit des Algorithmus nicht mehr zum Tragen kommt (Bitlänge sechs). Die optimale Minimallänge liegt in den meisten

Tabelle 6.3.: Vergleich der optimalen Schwellwerte des textbasierten MiLe-Fingerprints

MiLe-Konfiguration	F1-Maß	Schwellwert
Bitlänge 4, Minimallänge 5	0,9822	0,2440
Bitlänge 4, Minimallänge 7	0,9821	0,2068
Bitlänge 3, Minimallänge 5	0,9801	0,2894
Bitlänge 3, Minimallänge 7	0,9787	0,2077
Bitlänge 2, Minimallänge 8	0,9782	0,2902
Bitlänge 5, Minimallänge 4	0,9758	0,2330
Bitlänge 5, Minimallänge 5	0,9754	0,1996
Bitlänge 6, Minimallänge 3	0,9733	0,3436
Bitlänge 6, Minimallänge 7	0,9661	0,1314

Fällen zwischen fünf und acht. Bei größerer Bitlänge kann die Minimallänge auch kleiner gewählt werden. Das optimale Ergebnis von 98,22 % wird bei einer Bitlänge von vier und einer Minimallänge von fünf erzielt.

Die Bitlänge und insbesondere die gewählte Minimallänge haben direkten Einfluß auf den als optimal berechneten Schwellwert für die Abdeckung. Tabelle 6.3 zeigt die optimalen Schwellwerte für die zwei besten Konfigurationen jeder gewählten Bitlänge. Je größer die Mindestlänge, desto geringer ist die berechnete Abdeckung zwischen zwei Dokumenten. Wenn durch eine hohe Bitlänge zusätzlich (gewollte) Kollisionen vermieden werden, verringert sich diese noch stärker. Wenn hingegen eine sehr kleine Bitlänge gewählt wird, steigt die durchschnittlich berechnete Abdeckung an und dementsprechend der Schwellwert. In den meisten Parametrisierungen liegt der Schwellwert jedoch zwischen 20 % und 30 %.

Auswertung des objektbasierten MiLe-Fingerprints auf dem LIS.KOM-Korpus

Für die Auswertung des objektbasierten MiLe-Fingerprints ergaben sich 244 gültige und 46 als Vorlage eingestufte Variantenrelationen zwischen PowerPoint-Dokumenten. Wiederum kann die Evaluation somit in ein einfaches Kategorisierungsproblem überführt werden. Für jedes Dokumentenpaar einer Relation wird durch den Ansatz - also den objektbasierten MiLe-Fingerprint - ein Wertepaar berechnet, das die Abdeckung der Dokumente angibt. Zudem besitzt jedes Paar eine Zielkategorie. Es wurde ein zehnfacher Kreuzvalidierungsansatz für die Auswertung gewählt. Auf der Trainingsmenge wurde jeweils pro Ansatz der optimale Schwellwert für die Unterteilung in die beiden Kategorien berechnet, auf die Testmenge angewandt und anhand der gegebenen Zielkategorien das F1-Maß bestimmt. Die Ergebnisse wurden über 100 Durchgänge mit unterschiedlicher Aufteilung gemittelt. Da die Kategorien im Korpus nicht gleichverteilt vorhanden sind, ergibt sich eine Baseline, also eine Referenzgüte für das F1-Maß von 84,1 % für eine richtige Einordnung der Kategorie. Diesen Wert erreicht ein Algorithmus, wenn er einfach jede Relation für gültig erklärt, da gültige Relationen in dem getesteten Korpus viel häufiger sind als ungültige. Der Fingerprint wurde in den vier in Kapitel 6.2.2 beschriebenen Ausprägungen für die Gewichtung der Objekte sowie für unterschiedliche Konfigurationen des Basis-MiLe getestet und verglichen. Jede der Gewichtungsmethoden wurde dabei jeweils mit einer Bitlänge von 4 und 5 und einer

Minimallänge von 1 bis 20 gemessen. Bei den Gewichtungsmethoden handelt es sich um die einfache Gewichtung ("Einfach"), die Gewichtung anhand der Folienfläche ("Folienfläche"), die Gewichtung anhand der Gesamtfläche der Shapes auf einer Folie ("Gesamtfläche") und einer Mischung aus den beiden letzteren ("Gemischt").

Abbildung 6.8 zeigt die Ergebnisse der verschiedenen Gewichtungsmethoden für eine MiLe-Bitlänge von 4 in Abhängigkeit von der Minimallänge im Vergleich zur Baseline. Alle Gewichtungsmethoden liefern Ergebnisse von weit über 90 % für das F1-Maß. Das F1-Maß ist hierbei der entsprechend der Verteilung der Muster über die beiden Kategorien gewichtete Durchschnitt der F1-Maße beider Kategorien. Es zeigt sich, dass auf diesem Korpus das einfachste Gewichtungsmaß im Durchschnitt die besten Ergebnisse liefert. Den optimalen Wert von 97,2 % richtige Einordnung erreicht dieses Maß bei einer Minimallänge von 4. Die vergleichsweise schwächste Qualität weist die Gewichtungsmethode auf, die sich an der Gesamtfläche der Shapes einer Folie orientiert ("Gesamtfläche"). Des Weiteren wird ersichtlich, dass im Vergleich zu reinen Textdokumenten, die Kategorisierungsqualität auch für hohe Minimallängen noch sehr hoch ist. Das liegt daran, dass bei PowerPoint-Dokumenten sehr viele Shapes auf einer Folie vorhanden sein können, die sich bei Wiederverwendung in der entsprechenden Länge im Fingerprint wiederfinden. Durch eine Gewichtung und dadurch Replikation von einzelnen Hashwerten im Fingerprint, wird dieser Effekt zusätzlich verstärkt.

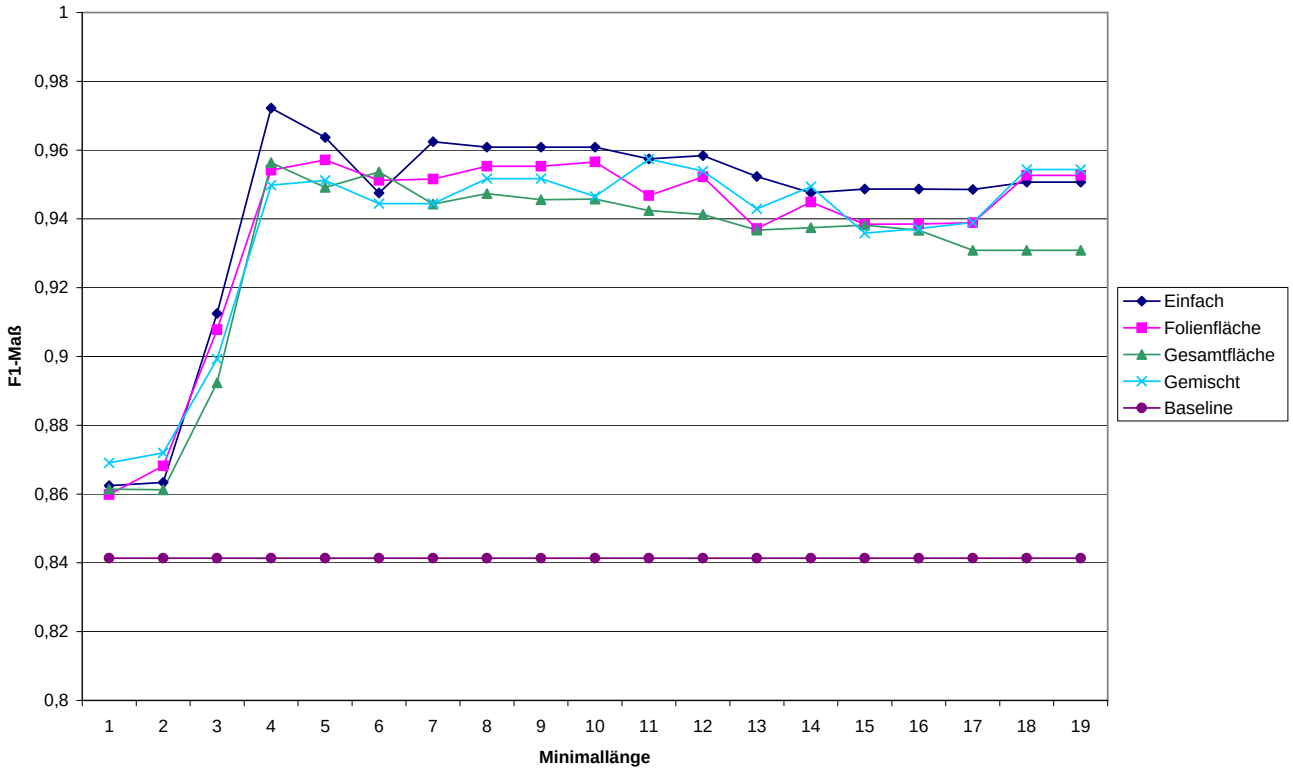

Abbildung 6.8.: Vergleich der Gewichtungsmethoden für den objektbasierten MiLe-Fingerprint für eine Bitlänge von 4

Bei Erhöhung der Bitlänge des zu Grunde liegenden MiLe-Fingerprints auf 5 Bit sind lediglich geringe Veränderungen im Vergleich zu 4 Bit erkennbar. Während der optimal erreichte Wert leicht sinkt, steigt die durchschnittliche Qualität aller Ansätze bis auf den "Gesamtfläche"-Ansatz. In dieser Konfiguration

wird noch stärker deutlich, dass dieser Ansatz den anderen Methoden unterlegen ist, auch wenn er die Baseline dennoch erheblich verbessert. Die schwächere Leistung dieser Gewichtungsmethode ist darin begründet, dass sich das Hinzufügen oder Entfernen eines Shapes auf die Gewichte aller anderen Shapes einer Folie auswirkt und somit auch auf deren Fingerprint-Sequenzen. Die Werte für die übrigen Ansätze liegen fast grundsätzlich zwischen 94 % und 96 %. Dies bedeutet, dass in der Praxis auf diesem Korpus für eine Bitlänge von 5 Bit und eine Minimallänge zwischen 3 und 18 eine Variantenrelation in >94 % der Fälle der richtigen Klasse zugeordnet - also eine Template-Variantenrelation von einer gültigen unterschieden wird. Die Ergebnisse machen auch deutlich, dass ein Bitlänge von 4 ausreichend ist.

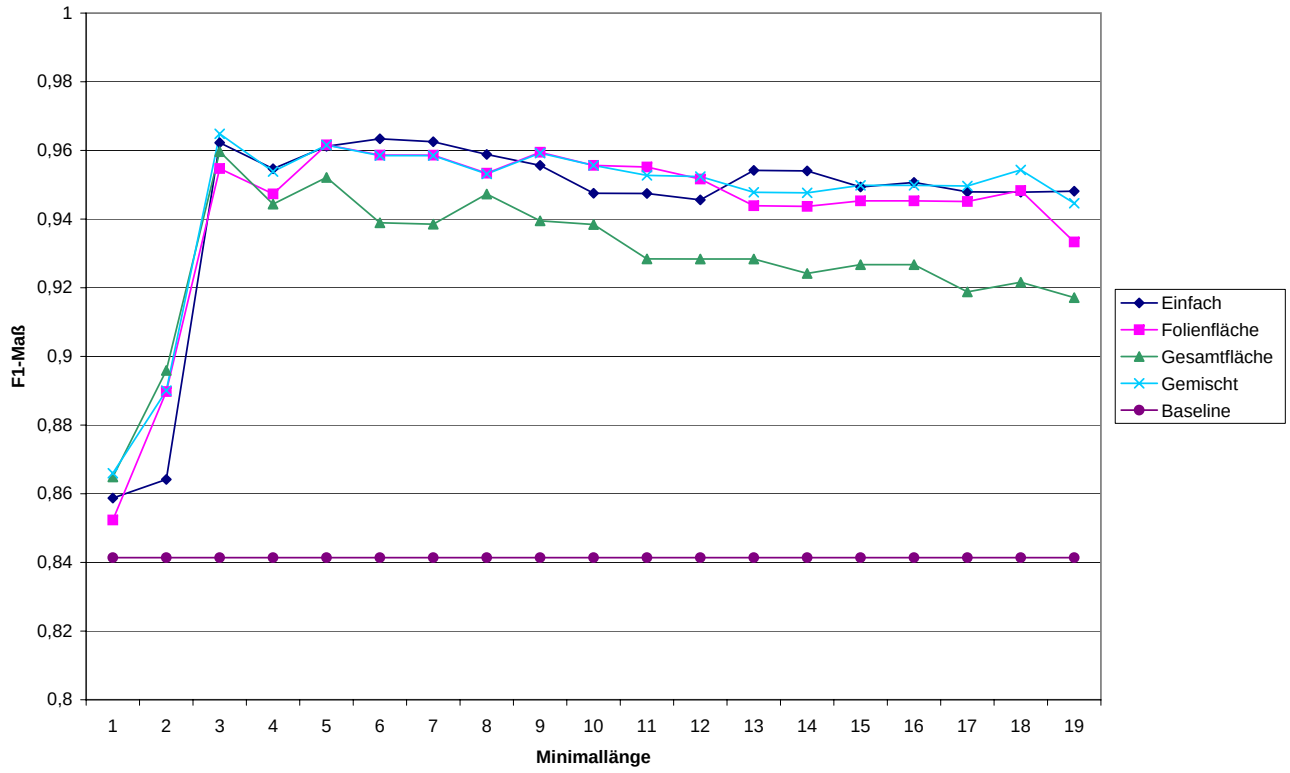

Abbildung 6.9.: Vergleich der Gewichtungsmethoden für den objektbasierten MiLe-Fingerprint für eine Bitlänge von 5

Neben der Qualität der Ansätze selbst, ist darüber hinaus noch eine Betrachtung der jeweils errechneten Grenzwerte des Abdeckungsmaßes für eine Zuordnung in die beiden Kategorien interessant. Tabelle 6.4 gibt eine Übersicht über die jeweils optimale Konfiguration für jede Gewichtungsmethode bei einer Bitlänge von 4 und 5 Bit und jeweils den für diese Konfiguration errechneten Schwellwert. Der Schwellwert wurde als Durchschnitt über alle während eines Testlaufs errechneten Schwellwerte berechnet. Es zeigt sich, dass der optimale Schwellwert generell zwischen 20 % und 40 % anzusetzen ist. Je nach Gewichtungsmethode und Minimallänge kann eine höhere oder geringere Abdeckung zwischen demselben Dokumentenpaar berechnet werden. Je höher die Minimallänge, desto geringer ist im Allgemeinen auch die berechnete Ähnlichkeit, da übereinstimmende Sequenzen, die kürzer als die Minimallänge sind nicht berücksichtigt werden. Dadurch ergibt sich in der Regel auch ein geringerer Schwellwert für die Unterscheidung. Bei einer kurzen Minimallänge (3 oder 4) ergibt sich auch ein höherer Schwellwert.

Tabelle 6.4.: Vergleich der optimalen Schwellwerte des objektbasierten MiLe-Fingerprints

Konfiguration / Ansatz	**F1-Maß**	**Durchschnittlicher Schwellwert**
Einfach (Bitlänge 4, Minimallänge 4)	0,9722	0,3289
Einfach (Bitlänge 5, Minimallänge 6)	0,9634	0,1979
Folienfläche (Bitlänge 4, Minimallänge 5)	0,9571	0,2860
Folienfläche (Bitlänge 5, Minimallänge 5)	0,9616	0,3074
Gesamtfläche (Bitlänge 4, Minimallänge 4)	0,9564	0,3547
Gesamtfläche (Bitlänge 5, Minimallänge 3)	0,9597	0,3892
Gemischt (Bitlänge 4, Minimallänge 11)	0,9574	0,2593
Gemischt (Bitlänge 5, Minimallänge 3)	0,9648	0,4067

Beim Einsatz des Algorithmus zur Erkennung der Gültigkeit von Variantenrelationen sollte dies bei der Konfiguration berücksichtigt werden.

Bei Untersuchung der Fälle, in denen der Ansatz eine falsche Gültigkeitskategorie zurückliefert, ist ersichtlich, dass es sich hierbei zum Großteil entweder um Grenzfälle handelt, oder um Fälle, in denen eine Berechnung der Abdeckung auf Wortebene keine gültigen Ergebnisse liefern kann. Grenzfälle entstehen, wenn sich zwei Varianten eines Dokuments inhaltlich in nur sehr geringem Umfang überschneiden, es sich laut der in Kapitel 8.1.2 gegebenen Bewertungskriterien aber dennoch um eine gültige Variantenrelation handelt. Dies ist zum Beispiel dann der Fall, wenn zwischen einer frühen Version eines Dokuments und dem fertigen Dokument kaum Varianten angelegt wurden. So entsteht eine Variantenbeziehung zwischen der sehr frühen und der fertigen Version eines Wissensdokuments, die als gültig anzusehen ist, obwohl die Dokumente sich inhaltlich kaum überschneiden. Eine syntaktische Berechnung der Abdeckung versagt andererseits in Fällen, in denen eine syntaktische Ähnlichkeit nicht gegeben ist, es sich aber aufgrund inhaltlicher Verwandtschaft dennoch um eine Variante handelt. Dies ist zum Beispiel bei Übersetzungen von Dokumenten, wie sie stellenweise im LIS.KOM-Korpus enthalten sind, der Fall.

6.4 Zusammenfassung und Beurteilung

Die durchgeführten Auswertungen haben gezeigt, dass mit MiLe ein flexibler, und ressourcenschonender und performanter Fingerprinting-Algorithmus entwickelt wurde, der bessere Ergebnisse liefert als existierende Ansätze. Insbesondere die Robustheit gegenüber kleinen Änderungen und die Berechnung des Abdeckungsmaßes auf Token- (also Wort-)Basis und nicht auf Shingle-Basis grenzt den Algorithmus von

existierenden Lösungen ab. Im Folgenden werden die Ergebnisse zusammengefasst und hinsichtlich ihrer Bedeutung für das gegebene Anwendungsszenario als auch weitere mögliche Einsatzszenarien beurteilt.

6.4.1 MiLe im LIS.KOM-Framework

Das Einsatzszenario, für das der Algorithmus entwickelt wurde, ist die automatisierte Beurteilung der Validität von Variantenrelationen im LIS-KOM-Framework. In einem solchen Szenario werden die Fingerprints von zwei in Beziehung stehenden Dokumenten paarweise verglichen. Der Algorithmus muss also einen paarweisen Vergleich in Echtzeit durchführen können. Zusätzlich soll der vom Fingerprint benötigte Speicherbedarf so gering wie möglich sein. Diese Anforderungen erfüllt der entwickelte MiLe-Algorithmus. Die Evaluationen auf verschiedenen Korpora haben gezeigt, dass der MiLe-Ansatz zudem bessere Ergebnisse liefert als bekannte Verfahren. Es wurde ebenso gezeigt, dass der Algorithmus auch auf objektbasierte Dokumente anwendbar ist. Beide Formen des Algorithmus liefern auf den im LIS.KOM-Framework gesammelten Daten sehr gute Ergebnisse. Es konnte gezeigt werden, dass sie gut für den Einsatzzweck, für den sie entwickelt wurden, geeignet sind. MiLe erfüllt somit die gegebenen Anforderungen.

Ein weiteres Einsatzszenario des Ansatzes innerhalb des LIS.KOM-Frameworks liegt darin, nicht nur bestehende Relationen zu validieren, sondern auch Relationen aus bestehenden Dokumenten zu extrahieren. Dies kann zum Beispiel sinnvoll sein, wenn ein Dokument von einem nicht überwachten System in das LIS.KOM-Framework aufgenommen wird. Auch hierfür eignet sich der Ansatz gut. Da in einem solchen Fall sehr viel höhere Laufzeitanforderungen an einen Algorithmus gestellt werden, da nicht mehr lediglich ein Eins-zu-Eins Vergleich von Dokumenten erfolgt, sondern ein neu ins System kommendes Dokument mit allen bestehenden verglichen werden muss, empfiehlt sich hier der Einsatz einer entsprechenden Datenstruktur (z.B. eines invertierten Index). MiLe ist jedoch auch in einem solchen Fall in der Lage, bei vergleichbarer Laufzeit bessere Ergebnisse als existierende Algorithmen zu liefern.

6.4.2 Weitere Anwendungsmöglichkeiten

Neben der in dieser Arbeit angestrebten Anwendung von MiLe als Methode zur Validierung bestehender Beziehungsinformationen, kann der Ansatz auch in anderen Szenarien zum Einsatz kommen. Die Erkennung von Wiederverwendung, was die Aufgabe von MiLe ist, ist die Basis für viele Anwendungen in unterschiedlichen Bereichen. Dazu zählt beispielsweise die Erkennung von Plagiaten. Dabei geht es vordergründig darum, zu erkennen ob Inhalte eines Dokuments aus fremden Quellen wiederverwendet wurden und wenn ja aus welchen. In einem nächsten Schritt muss dann entschieden werden, ob es sich bei der Wiederverwendung um ein Plagiat oder ein rechtmäßiges Zitat handelt. Einige der in diesem Kapitel beschriebenen verwandten Algorithmen kommen auch für die Erkennung von Plagiaten zum Einsatz (siehe zum Beispiel [Clo03],[PMP02],[BCR09]). Besonders häufig werden hierbei K-Gram oder der im vorherigen Kapitel beschriebene GST-Algorithmus verwendet. Während GST ein schwaches Laufzeitverhalten zeigt, hat K-Gram den Nachteil, dass mit seiner Hilfe Wiederverwendung zwar erkannt, jedoch nicht ohne weiteres innerhalb eines Dokumentes lokalisiert werden kann. Hierzu müsste zusätzlich zu jedem Shingle ein Offset gespeichert werden, was den Speicherbedarf wiederum erhöht. Anhand

eines MiLe-Fingerprints kann ein Plagiat erkannt und, abhängig von den verwendeten Vorverarbeitungsschritten, lokalisiert werden, ohne weitere Daten speichern zu müssen. Ein weiterer großer Vorteil des MiLe-Fingerprints ist, dass er eine kompakte Repräsentation des zu Grunde liegenden Dokuments darstellt. Dadurch kann eine Plagiatserkennung für ein Dokument allein anhand des MiLe-Fingerprints und ohne Zugriff auf den Inhalt des Dokuments selbst durchgeführt werden. Dies ist in Szenarien, in denen Datenschutz ein wichtiger Faktor ist, ein großer Vorteil.

Eine weitere Einsatzmöglichkeit ist das Tracking von Informationen. In [KcCT09] sowie [MBC+05] werden Techniken zur Erkennung von Wiederverwendung eingesetzt, um den Informationsfluß von News-Seiten beziehungsweise Blogs nachzuverfolgen und darstellbar zu machen, woher bestimmte Informationen stammen und welchen Weg sie durch das Web genommen haben. Auch hierfür ist MiLe geeignet. Darüber hinaus kann MiLe auch für die Suche eingesetzt werden, beispielsweise um verwandte Dokumente schnell auffindbar zu machen.

Teil IV.

Implementierung, Evaluation und Zusammenfassung

7 Implementierung des LIS.KOM-Frameworks

Im folgenden Kapitel wird die Implementierung des in Kapitel 4 konzipierten LIS.KOM-Systems vorgestellt. Es erfolgt zunächst ein Überblick über die Architektur des Gesamtsystems, bevor die einzelnen Komponenten zur Erfassung, Verwaltung und Nutzung von Lebenszyklusinformationen im Einzelnen dargestellt werden. Auch hier stehen die Erfassung und Verwaltung und nicht die Nutzung der Informationen und ihre Umsetzung im Fokus. Dennoch wurden prototypisch zwei unterschiedlich Möglichkeiten zur Nutzung der erfassten Informationen umgesetzt.

7.1 Gesamtarchitektur

Wie im Konzept vorgesehen, wurde das LIS.KOM-Framework als Client-Server-System umgesetzt. Lebenszyklusinformationen werden auf verschiedenen Clients erfasst und auf einem zentralen Server verwaltet. Im Folgenden wird der Weg der Informationen durch das LIS.KOM-System von der Erfassung bis zur Nutzung beschrieben. Er kann anhand von Abbildung 7.1 nachvollzogen werden. Er beginnt links unten mit der Erfassung der Informationen durch Plug-ins in verschiedenen Applikationen (Erfassung), führt durch den LIS.KOM-Client, der als lokaler Cache dient, auf den LIS.KOM-Server, wo die Informationen aller Clients zentral gespeichert werden (Verwaltung) und endet links oben mit der Nutzung der Informationen, die wiederum entweder in Form von Plug-ins in Applikationen integriert oder in Form von eigenständigen Applikationen umgesetzt werden kann (Nutzung).

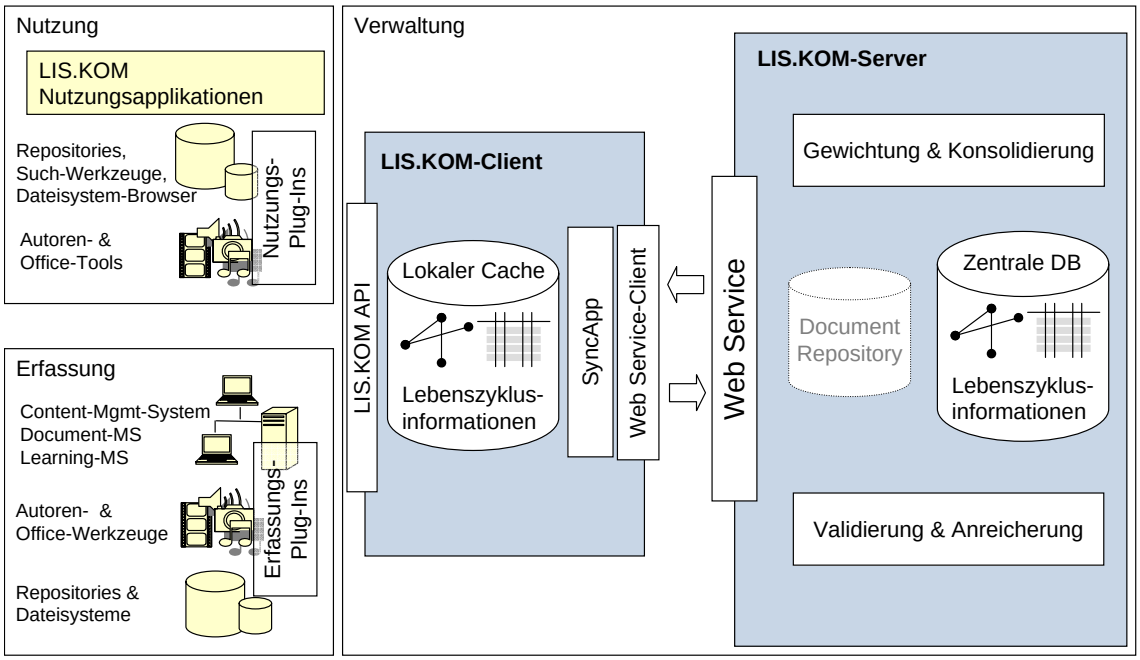

Abbildung 7.1.: Die Architektur des LIS.KOM-Frameworks

Die Erfassung der Informationen erfolgt innerhalb der Applikationen, in denen die Informationen entstehen und wird wie vorgesehen mit Hilfe von Plug-ins bewerkstelligt. Die Plug-ins erfassen die Informationen und senden sie an einen lokal auf dem Client vorhandenen Cache (LIS.KOM-Client), wo sie zwischengespeichert werden und eine erste Validierung erfolgt (siehe 7.3.2). Die Nutzung des LIS.KOM-Clients als lokalen Cache hat den Vorteil, dass nicht grundsätzlich eine Verbindung zum LIS.KOM-Server bestehen muss. Zu gegebenen Zeitpunkten werden die im lokalen Cache verfügbaren Informationen über einen Webservice mit dem LIS.KOM-Server synchronisiert. Auf dem Server erfolgt eine weitere Verarbeitung zum Beispiel durch Konsolidierung, Anreicherung und Gewichtung (siehe 4.2.4). Neben den Lebenszyklusinformationen können auch die Dokumente selbst auf den Server übertragen werden. Dies kann geschehen, wenn der LIS.KOM-Server beispielsweise einen direkten Zugriff auf die Dokumente selbst für die Nutzung ermöglichen soll, oder die Dokumente für die Evaluation des Systems benötigt werden (siehe Kapitel 8). Hierzu stellt der Server ein dateisystembasiertes Repository zur Verfügung, aus dem die Dokumente mit Hilfe des Webservice heruntergeladen werden können. Die aufbereiteten Informationen können wiederum an den lokalen Cache gesendet werden, um dort von Nutzungsanwendungen verwendet zu werden. Es besteht jedoch auch die Möglichkeit die Informationen direkt mit Hilfe des Webservice vom LIS.KOM-Server zu beziehen. Dies setzt jedoch voraus, dass zum Zeitpunkt der Nutzung eine Verbindung vom Client zum Server besteht. Ist dies nicht der Fall, fungiert der LIS.KOM-Client als Cache. Die Lebenszyklusinformationen können einerseits durch Plug-ins in bestehende Applikationen zur Nutzung eingebunden werden, um deren Funktion zu erweitern. Alternativ kann eine unabhängige Applikation die Daten auch direkt nutzen. Die folgenden Unterkapitel sind anhand der drei Hauptfunktionalitäten des LIS.KOM-Frameworks in Erfassung (7.2), Verwaltung (7.3) und Nutzung (7.4) aufgeteilt.

7.2 Erfassungskomponenten

Für die Erfassung von Beziehungsinformationen aus dem Lebenszyklus von Wissensdokumenten wurden im Rahmen dieser Arbeit drei Erfassungskomponenten entwickelt, jeweils ein Plug-in für Microsoft PowerPoint und Microsoft Word und eine Applikation, die im Hintergrund läuft und Informationen aus dem Dateisystem erfasst. Die Wahl der Komponenten ergab sich aus der Analyse des in Kapitel 2 modellierten Lebenszyklus. Des LIS.KOM-Framework sollte sowohl PowerPoint- als auch Word-Dokumente unterstützen. Um den Lebenszyklus dieser Dokumente komplett abdecken zu können, ist ein Erfassung-Plug-in in der jeweils zugrundeliegenden Applikation jedoch nicht ausreichend. Zwar erfolgt sowohl die Überarbeitung als auch die Nutzung auf Desktop-Rechnern zumeist innerhalb dieser Applikationen, die Bereiststellungs- und Zugriffsphase wird dadurch jedoch nicht abgedeckt. Diese spielt sich auf den genannten Systemen zumeist im lokalen Dateisystem ab, weshalb für dieses eine zusätzliche Erfassungs-Komponente (der Filesystem-Monitor) umgesetzt wurde. Jedes der Plug-ins deckt dabei eine der in Kapitel 4.1 beschriebenen prinzipiellen Herangehensweisen an die Erfassung ab. Das PowerPoint-Plug-in verfolgt alle relevanten Aktionen des Nutzers und hält so die gesammelten Informationen immer synchron mit dem Bearbeitungszustand des Dokuments, während das Word-Plug-in Beziehungen erfasst und diese zu bestimmten Zeitpunkten mit Hilfe von Validierungsmethoden mit dem Zustand des Dokumentes abgleicht. Im Folgenden werden zunächst die Gemeinsamkeiten beider Plug-ins beschrieben, die in der Erfassung der Relationen selbst besteht, bevor die drei Komponenten und ihre Besonderheiten

im Einzelnen erläutert werden. Schließlich wird das Konzept bezüglich der Übertragbarkeit auf andere Applikationen beurteilt.

7.2.1 Gemeinsame Aspekte der Plug-ins

Eine allgemeine Herausforderung bei der Erfassung von Lebenszyklusinformationen in PowerPoint sowie in Word ist, dass die Schnittstellen, die diese Applikationen zu Verfügung stellen, nicht dafür konzipiert sind, die Aktionen eines Nutzers zu überwachen. Sie sind ursprünglich dafür konzipiert, die Applikationen zu automatisieren, um so mit Hilfe von Programmcode Dokumente oder Präsentationen erstellen oder anpassen zu können. Daher muss für die Erfassung der notwendigen Informationen in einigen Fällen auf Informationen aus dem Betriebssystem zurückgegriffen, oder mit Hilfe von Heuristiken und Annahmen gearbeitet werden. Beide Plug-ins wurden als Add-ins auf Applikationsebene in C# [ECM09] umgesetzt. Das bedeutet, dass die Plug-ins an die Applikation gekoppelt sind und jedes Mal, wenn die Applikation geladen wird, automatisch ebenfalls geladen werden. Alternativ dazu könnte eigene Programmierlogik auch auf Dokumenten- oder Template-Ebene in Office-Applikationen integriert werden, diese ist jedoch dann an die jeweiligen Templates und Dokumente gebunden. Da das LIS.KOM-Framework beliebige Dokumente und Templates in den gegebenen Applikationen unterstützen soll, kommt diese Möglichkeit nicht in Frage.

Für die eigentliche Erfassung einer Relation und der zugehörigen Informationen kommen in Word wie in PowerPoint größtenteils die gleichen Techniken zum Einsatz. Das Plug-in unterstützt jeweils die Erfassung von Varianten-, Element- und Aggregationsrelationen, da diese bei den gegebenen Dokumenttypen am häufigsten auftreten. Variantenrelationen werden von beiden Plug-ins erfasst, indem überwacht wird, wann eine neue Instanz eines Dokuments gespeichert wird. Dies lässt sich zumeist über Ereignisse, die von der API der Applikation bereitgestellt werden, bewerkstelligen. Alternativ hierzu können Varianten - wie in Kapitel 4.1 beschrieben - noch durch anderen Aktionen, zum Beispiel durch Kopieren im Dateisystem entstehen. Diese werden mit Hilfe des Filesystem-Monitors erfasst (siehe Kapitel 7.2.4).

Aggregations- und Elementrelationen werden erfasst für alle Aktionen, die das Einfügen eines in einem anderen Dokument kopierten Artefakts in ein Word- oder PowerPoint-Dokument zur Folge haben. Die Quelle des Inhalts kann also auch ein Nicht-PowerPoint- oder Nicht-Word-Dokument sein. Im weiteren Verlauf wird das Dokument, in dem etwas kopiert wird, als Quelldokument bezeichnet, während das Dokument, in das eingefügt wird, Zieldokument genannt wird. Die Erfassung läuft in zwei Schritten ab: Während des Kopiervorgangs werden Informationen über das Quelldokument und gegebenenfalls Quellartefakte erfasst, und nach dem Einfügen des Inhalts werden die Informationen über das Zieldokument und Zielartefakte erhoben.

Im Folgenden werden zunächst die Ereignisse, die zur die Erfassung von Beziehungsinformationen verfolgt werden müssen, erläutert. Nachfolgend wird beschrieben, welche Dokumenttypen als Quelldokumente einer Relation unterstützt werden und schließlich werden die erfassten Informationen selbst zusammengefasst.

Notwendige Ereignisse

Um eine Relation zu erfassen ist es folglich notwendig, zu erfahren, wenn Inhalte kopiert und wenn sie in ein überwachtes Dokument eingefügt werden. Um zu erkennen, ob Inhalt kopiert wurde, wird die Zwischenablage des Betriebssystems - das sogenannte Clipboard - überwacht. Bei einer Änderung des Clipboards, prüft das Plug-in, wo die Änderung erfolgt ist, also in welchem Dokument Inhalt kopiert wurde. Dieser Schritt erfolgt grundsätzlich, wenn eines der Plug-ins in Betrieb ist. Die erfassten Informationen werden im Speicher zwischengelagert bis der Inhalt in ein entsprechendes Dokument eingefügt wird oder ein neuer Kopiervorgang erfolgt. Um herauszufinden, wann Inhalt in ein Dokument eingefügt wurde kann zum Einen die Tastatur hinsichtlich des zum Einfügen verwendeten Kürzels "Strg + V" und zum Anderen die Benutzerschnittstelle der Applikation hinsichtlich ihrer Funktionen zum Einfügen von Inhalten aus der Zwischenablage überprüft werden. Neben diesen, für einen Kopiervorgang spezifischen Ereignissen, die vom Betriebssystem oder der API der überwachten Applikation zur Verfügung gestellt werden müssen, damit Relationen erfasst werden können, müssen auch noch verschiedene grundlegende Ereignisse und Informationen verfügbar beziehungsweise zugreifbar sein, um die Funktionalität der Erfassung zu gewährleisten. Diese werden im Folgenden erläutert.

- **Öffnen** eines Dokuments: Um Informationen zu einem Dokument erfassen zu können, muss das System über das Öffnen eines Dokuments notifiziert werden. Dies geschieht im Fall von Word und PowerPoint mit Hilfe der Plug-ins, die automatisch beim Öffnen eines Dokuments geladen werden.

- **Schließen** eines Dokuments: Ebenso wie das Öffnen eines Dokuments muss es den Plug-ins auch möglich sein, zu erkennen, wann ein Dokument geschlossen wurde. Somit kann die Erfassung von Informationen und Überwachung der Relationen dieses Dokuments unterbrochen werden. Gegebenenfalls können gesammelte Informationen aus dem Speicher in den lokalen Cache oder auf den LIS.KOM-Server persistiert werden. Auch dieses Ereignis wird im Fall von Word und PowerPoint wie auch in den meisten anderen Fällen von der API der Applikation bereitgestellt.

- **Aktivität** eines Dokuments: Wenn es möglich ist, mehrere verschiedene Dokumente in einer Applikation gleichzeitig geöffnet zu haben, muss das Plug-in in der Applikation wissen, welches der geöffneten Dokumente gerade aktiv ist. Diese Information kann wiederum zumeist mit Hilfe der API der Applikation abgerufen werden, ansonsten besteht die Möglichkeit die Information über Funktionen des Betriebssystems zu erfassen.

- **Speichern** eines Dokuments: Auch das Speichern eines Dokuments ist ein relevantes Ereignis. Beim Speichern werden Änderungen an einem Dokument persistiert und durch diese Änderungen hervorgerufene Relationen können gespeichert werden. Zudem gibt es unterschiedliche Aktionen auf dem Dokument selbst, wie zum Beispiel die Vergabe der ID, die es erfordern, dass das Dokument gespeichert wird.

- **Erstellen** eines Dokuments: Zumeist wird durch das Erstellen eines neuen Dokument ein Ereignis der API der Applikation getriggert, das Ereignis kann jedoch auch mit Hilfe des Dateisystems überwacht werden.

- Aktionen auf Dokumenten im **Dateisystem**: Auch Aktionen auf Dokumenten, die außerhalb der überwachten Applikation erfolgen, müssen in Betracht gezogen werden. Diese Aufgabe wird vom Filesystem-Monitor abgedeckt.

Unterstützte Dokumenttypen

Die für eine Relation notwendigen Informationen werden abhängig vom Typ des Quelldokuments auf unterschiedliche Weise erfasst. Dabei wird eine Vielzahl von Dokumenttypen unterstützt:

- Erfolgt der Kopiervorgang **innerhalb derselben Applikation**, die auch das erfassende Plug-in beinhaltet (also entweder Word oder PowerPoint), so können die notwendigen Informationen sehr einfach mit Hilfe des Plug-ins erfasst werden.
- Wird Text oder HTML aus einem **Browser** kopiert, so können die benötigten Information über das Quelldokument - also die URL der Webseite - sowie der kopierte Text aus dem Clipboard gewonnen werden.
- Wenn **andere Inhalte im Browser** kopiert werden, wie zum Beispiel Bilder oder Text aus innerhalb des Browsers geöffneten PDF-Dateien, kann die geöffnete URL nicht mit Hilfe des Clipboards erfasst werden. Hierfür wurde ein Screen-Scraping-Ansatz realisiert. Dabei werden alle möglichen Felder des Browsers zugegriffen und daraufhin überprüft, ob sie die momentan geöffnete URL enthalten.
- Wenn Dateien aus dem **Dateisystem** kopiert werden, können die notwendigen Informationen - in diesem Fall der Pfad und Dateiname der kopierten Datei - ebenfalls aus dem Clipboard extrahiert werden. Dieser Anwendungsfall tritt insbesondere bei Aggregationsrelationen auf.
- Bei **anderen lokalen Applikationen**, wie zum Beispiel Textverarbeitungsprogrammen, Modellierungswerkzeugen, Zeichen- oder Grafikprogrammen, kann in den meisten Fällen der Name der geöffneten Datei mit Hilfe der Betriebssystem-API aus dem Fenstertitel der Applikation ausgelesen werden. Neben dem Namen wird noch der vollständige Pfad der Datei benötigt. Dieser kann mit Hilfe eines Systemordners gefunden werden, der verwendet wird, um die zuletzt geöffneten Dateien zu speichern ('Recent Folder'). Die meisten Applikationen nutzen diesen Ordner und legen eine Referenz zu jeder kürzlich geöffneten Datei in diesen Ordner.

Die Erfassung deckt folglich einen sehr großen Teil der möglichen Quelldokumente eines Kopiervorgangs ab.

Erfasste Informationen

Zunächst erfolgt die Erfassung der Informationen über das Quelldokument und gegebenenfalls Quellartefakte während des Kopiervorgangs. Je nach Quelldokument und Relationstyp, werden unterschiedliche Informationen über das Dokument selbst und das kopierte Artefakt benötigt (siehe Kapitel 4.2.3). Grundsätzlich wird der Pfad beziehungsweise die URL des Quelldokuments benötigt. Besitzt das Dokument bereits eine ID, so wird diese verwendet, andernfalls wird eine neue vergeben (siehe Kapitel

7.3.1). Webseiten wird zwar eine ID zugewiesen, ihre Identifikation erfolgt jedoch über die URL. Der Typ des Quelldokuments ergibt sich meist aus der Endung des Dateinamens und als Besitzer wird bei neu angelegten Dokumenten der Nutzer, der das Dokument geöffnet hat, eingetragen. Da eine Aggregationsrelation für das Quelldokument kein Artefakt besitzt, sind die Informationen hierfür vollständig. Bei Elementrelationen werden je nach Dokumenttyp weitere Informationen benötigt. Welche Informationen jeweils erfasst werden können, ist dabei auch vom Typ des kopierten Artefakts abhängig:

- Wird aus PowerPoint kopiert, so wird die ID der Folie, von der kopiert wurde, erfasst. Zusätzlich kann noch eventuell kopierter Text sowie die IDs gegebenenfalls einzeln kopierter Elemente erhoben werden. Werden mehrere Folien auf einmal kopiert, werden auch mehrere Relationen erzeugt.
- Beim Kopieren aus Word wird der Text erfasst. Wenn neben Text noch andere Objekte, wie Bilder oder Grafiken kopiert wurden, wird für diese eine interne ID vergeben und an dem Objekt innerhalb des Dokumentes selbst gespeichert. Anhand dieser kann das Objekt identifiziert werden.
- Bei allen anderen Applikationstypen, wird neben den Informationen über das Quelldokument noch der jeweils kopierte Text, beziehungsweise ein Hashwert für andere Objekte, erfasst.

Die Informationen über das Zieldokument, was stets ein PowerPoint beziehungsweise Word-Dokument ist, können erst erfasst werden, wenn der kopierte Inhalt in das Dokument eingefügt wurde. Eine Herausforderung hierbei ist, dass das Ereignis für das Einfügen von Inhalten in ein Dokument ausgelöst wird, kurz bevor der Inhalt tatsächlich eingefügt wurde. Somit können zu diesem Zeitpunkt zwar Informationen über das Zieldokument, welches dann schon aktiviert ist, erfasst werden, aber nicht über das Zielartefakt. Um dies dennoch zu ermöglichen wird ein Marker gesetzt, damit bei nächster Gelegenheit die fehlenden Informationen gewonnen werden können. Hierzu wird ein Ereignis verwendet, welches garantiert nach jedem Einfügen von Inhalten aus dem Clipboard auftritt - das "SelectionChanged" Ereignis. Bei jedem Auftreten dieses Ereignisses wird der Marker geprüft. Wenn er gesetzt ist, kann angenommen werden, dass es sich bei dem aktuell ausgewählten Objekt, beziehungsweise dem aktuell selektierten Textbereich, um das eingefügte Artefakt der aktuellen Relation handelt. Hierfür werden dann je nach Applikation wiederum der Text und die entsprechenden IDs und Typen erhoben, um das Zielartefakt zu vervollständigen.

7.2.2 Das PowerPoint-Plug-in

Das PowerPoint-Plug-in verfolgt die Strategie der kompletten Nachverfolgung aller Nutzeraktionen, die sich auf erfasste Relationen auswirken können. Hierzu müssen neben den zuvor vorgestellten auch Aktionen erfasst werden, die dazu führen können, dass eine erfasste Relation ihre Validität verliert. Diese wurden bereits in Kapitel 4.1 erläutert (siehe Tabelle 4.1). In PowerPoint geschieht dies hauptsächlich durch Löschen von Elementen auf Folien oder durch Löschen ganzer Folien. Ein besonderes Augenmerk erfordert hierbei die gegebene Möglichkeit, Aktionen rückgängig zu machen ("Undo"), beziehungsweise rückgängig gemachte Aktionen wiederherzustellen ("Redo"). Dies kann sowohl das Einfügen von kopierten Inhalten als auch das Löschen von Elementen betreffen. Da die PowerPoint-API den Zugriff auf die Undo- und Redo-Stapel nicht unterstützt, muss ein eigener Handler für die Verwaltung der diesbezüglich relevanten Nutzeraktionen sorgen. Diesen Zweck erfüllt der *Stack-Handler* (siehe Abbildung 7.2), er

besteht aus zwei Stapeln, die Informationen zu getätigten Nutzeraktionen, sowie zu den davon jeweils betroffenen Relationen enthalten. Eine durchgeführte Aktion, die rückgängig gemacht werden kann, wie zum Beispiel das Löschen oder Einfügen einer Folie, wird auf dem Undo-Stapel abgelegt. Wird diese Aktion durch den Nutzer rückgängig gemacht, wird die zugehörige Information auf den Redo-Stapel verschoben und die Relation angepasst. Entsprechend wird verfahren, wenn die rückgängig gemachte Aktion wiederhergestellt wird. Da mehrere Aktionen in Folge rückgängig gemacht und wiederhergestellt werden können, ist es notwendig, die komplette Historie der Aktionen auf den Stapeln zu halten. Die Stapel funktionieren nach dem Last-In-First-Out-Prinzip - die zuletzt durchgeführte Aktion wird zuerst rückgängig gemacht und umgekehrt.

Aufgrund der Undo-Funktion darf eine einmal erfasste Relation selbst beim Löschen der betreffenden Elemente nicht vollständig aus dem lokalen Cache gelöscht werden. Beim Wiederherstellen könnten die Quellinformationen nicht vollständig reproduziert werden. Deshalb werden alle erfassten Relationen bis zum Schließen oder Speichern des Dokuments vorgehalten und lediglich als gelöscht markiert. Auch Relationen, die durch aktuelle Änderungen im Dokument, die noch nicht gespeichert wurden, entstanden sind, müssen entsprechend als "neu" markiert werden. Beim Speichern werden die neuen Relationen persistiert und gelöschte Relationen endgültig entfernt, während beim Schließen ohne zu speichern neue Relationen verworfen und gelöschte wiederhergestellt werden.

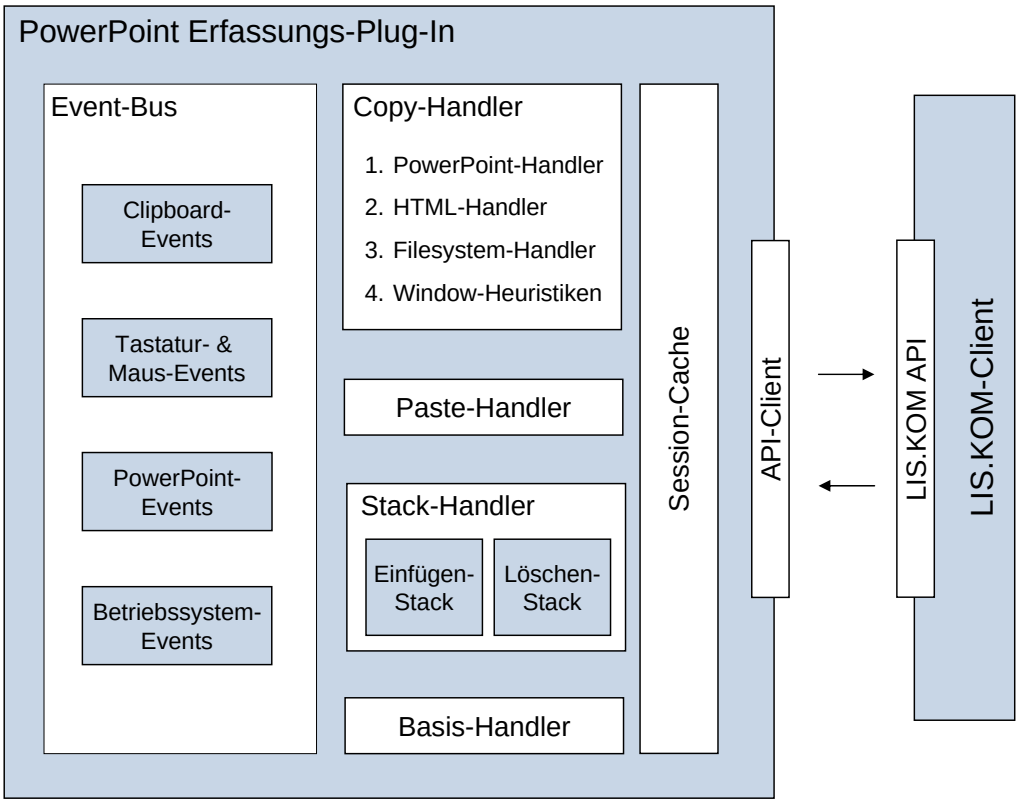

Abbildung 7.2.: Schematischer Aufbau des PowerPoint-Plug-ins

Abbildung 7.2 zeigt den Aufbau des PowerPoint-Plug-ins zur Erfassung von Lebenszyklusinformationen. Der Event-Bus verwaltet die Ereignisse aus den verschiedenen Quellen und leitet sie an die entsprechenden Komponenten weiter. Die Informationen zu einem aktuellen Kopiervorgang werden jeweils im

Session-Cache gespeichert und erst bei Vollendung der Aktion in den lokalen Cache des LIS.KOM-Client übertragen. Der Copy-Handler ist dabei wie beschrieben für die Erfassung der Informationen über das jeweilige Quelldokument zuständig. Die Rückverfolgung der Informationen erfolgt mit Hilfe des Stack-Handlers. Der Basis-Handler verwaltet grundlegende Ereignisse wie das Öffnen, Schließen und Speichern von Dokumenten. Der Paste-Handler verfolgt das Einfügen von Kopierten Inhalten in der in Abschnitt 7.2.1 beschriebenen Art und Weise.

7.2.3 Das Word-Plug-in

Die Erfassung der Relationen an sich funktioniert im Word-Plug-in auf dem selben Weg wie beim PowerPoint-Plug-in. Das Word-Plug-in verfolgt den Ansatz der regelmäßigen Validierung. Das heißt, die erfassten Relationen werden nicht ständig mit dem Zustand des Dokument synchron gehalten. Hierfür bietet die Word-API zu wenig Unterstützung, zum Beispiel in Form von entsprechenden Ereignissen. Ein weiteres Problem ist dadurch gegeben, dass im Gegensatz zu objektbasierten Dokumenten, wie PowerPoint-Präsentationen, textbasierte Dokumente eine geringe Strukturiertheit aufweisen. Deshalb können einzelne Relationen nicht auf bestimmte Objekte abgebildet werden, was eine vollständige Nachverfolgung erschwert. Die für ein Word-Dokument erfassten Element- und Aggregationsrelationen werden bei jedem Speichervorgang sowie wenn das Dokument geschlossen wird mit Hilfe der in Kapitel 5 beschriebenen Techniken validiert. Variantenrelationen werden in der implementierten Version nicht zur Laufzeit des Systems validiert.

Eine Besonderheit ergibt sich durch die Erfassung von Ereignissen, die das Speichern eines Dokuments betreffen. Während die PowerPoint-API Ereignisse auslöst, bevor und nachdem eine Präsentation gespeichert wurde, löst die Word-API lediglich vor dem Speichervorgang ein Ereignis aus. Da der Speicherprozess jedoch durch den Nutzer abgebrochen werden kann, kann nicht angenommen werden, dass in einem solchen Fall das Dokument wirklich gespeichert wird. Auch in diesem Fall hilft ein Merker, welcher in einem späteren Ereignis abgefragt wird, um die beim Speichern notwendigen Aktionen durchführen zu können.

Abbildung 7.3 zeigt den schematischen Aufbau des Word-Plug-ins. Der überwiegende Teil der Komponenten ist identisch mit denen des PowerPoint-Plug-ins. Die Relationen aus dem Session-Cache werden im Word-Plug-in jedoch lediglich bei Speichervorgängen an den LIS.KOM-Client übertragen und solange im Speicher vorgehalten. Bei der Übertragung werden die Relationen auch gleichzeitig validiert. Dies geschieht durch die Validierungskomponente. Objekte werden vordergründig mit Hilfe einer bei der Wiederverwendung vergebenen ID validiert. Ist das Objekt mit der ID in einem Dokument noch vorhanden, gilt die Relation als valide, wenn nicht, wird die Relation gelöscht. Durch inhaltliche Validierung mit Hilfe von Hashwerten, kann zusätzlich überprüft werden, ob ein Objekt verändert wurde. Textbasierte Elementrelationen werden mit Hilfe der in Kapitel 5 beschriebenen Techniken validiert. Dazu wird mit Hilfe des ShingleCloud-Algorithmus überprüft, ob der in der Relation gespeicherte Text noch im Zieldokument der Relation vorhanden ist.

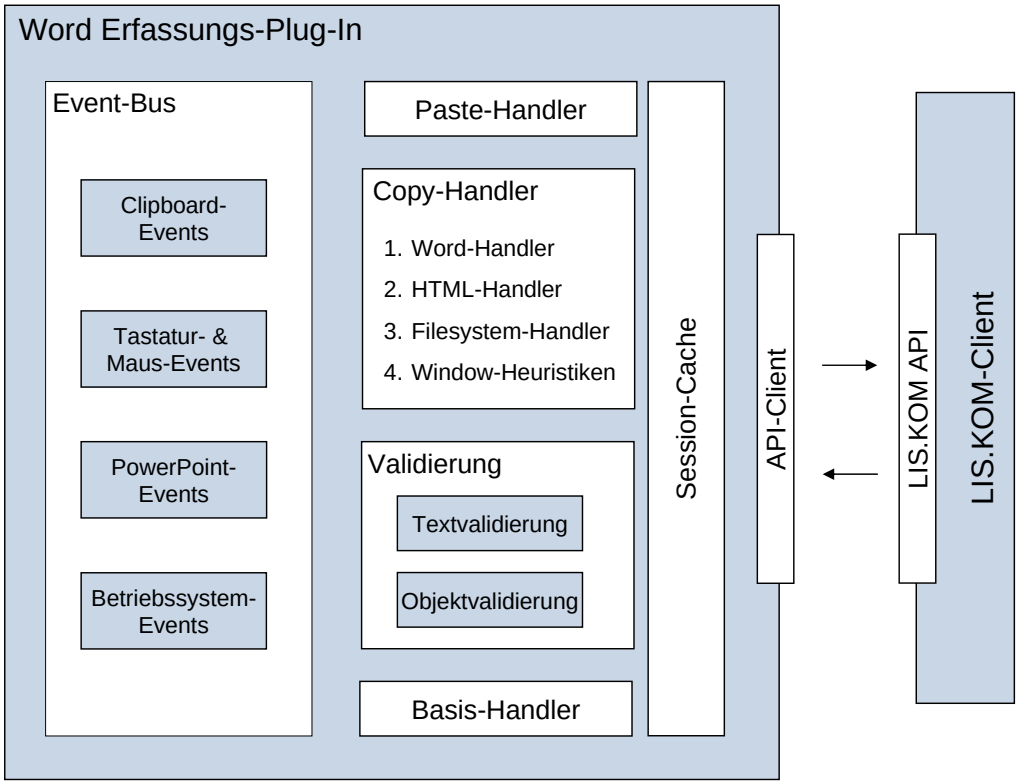

Abbildung 7.3.: Schematischer Aufbau des Word-Plug-ins

7.2.4 Der Filesystem-Monitor

Der Filesystem-Monitor ist kein Plug-in im eigentlichen Sinne, obwohl man ihn durchaus als "Plug-in des Betriebssystems" bezeichnen kann. Beim Filesystem-Monitor handelt es sich im Prinzip um eine Erfassungskomponente in Applikationsform. Die Applikation läuft ständig im Hintergrund und überwacht das lokale Dateisystem auf Aktivitäten, die zur Entstehung von Beziehungsinformationen führen können oder einen Einfluss auf vorhandene Beziehungsinformationen haben. Der Filesystem-Monitor besitzt keine Benutzeroberfläche außer eines Icons in der Windows-Task-Leiste, welches seine Deaktivierung erlaubt. Der Filesystem-Monitor deckt dadurch, dass er ständig im Hintergrund läuft, die Aktionen auf relevanten Dokumenten ab, die außerhalb der überwachten Applikationen Word und PowerPoint geschehen. So kann ein Dokument zum Beispiel im Dateisystem umbenannt, verschoben, kopiert oder gelöscht werden, was Auswirkungen auf bereits vorhandene Relationen haben kann (siehe Kapitel 4.1.1). Beim Löschen wird das Dokument durch den Filesystem-Monitor im lokalen Cache als gelöscht markiert, bei Umbenennung und Verschiebung werden die im lokalen Cache vorhandenen Daten entsprechend aktualisiert und bei Kopie wird eine neue ID vergeben und eine Variantenrelation erzeugt (siehe auch 7.3.1). Als gelöscht markierte Variantenrelationen werden nicht entfernt, um den Zusammenhang eines Variantenstrangs nicht zu verlieren. Durch die Markierung ist es dennoch möglich, die Relationen bei der Nutzung entsprechend auszublenden, oder die Relationen kurzzuschließen (siehe Kapitel 4.2.4).

7.2.5 Übertragbarkeit auf andere Applikationen

Für den Prototyp des LIS.KOM-Frameworks wurden zwei Erfassungs-Plug-ins entwickelt - für Microsoft Office PowerPoint sowie Microsoft Office Word. Um die Erfassung von Lebenszyklusinformationen in weiteren Applikationen zu ermöglichen, ist es notwendig, weitere Plug-ins zu entwickeln. Im Folgenden wird beschrieben, inwiefern die für die bestehenden Plug-ins entwickelten Konzepte und Komponenten auf weitere Applikationen übertragbar sind und welche Voraussetzungen Applikationen erfüllen müssen, damit eine Erfassung von Lebenszyklusinformationen innerhalb dieser Anwendungen möglich ist.

Die Konzepte der Validierung und der kompletten Nachverfolgung sind beide im Prinzip auf andere Applikationen übertragbar, haben jedoch unterschiedliche Anforderungen an die API der Applikation. Das auf Validierung basierende Konzept, wie es im Word-Plug-in umgesetzt ist, hat hierbei die geringeren Anforderungen. So ist lediglich die in Kapitel 7.2.1 beschriebene Mindestmenge an Ereignissen notwendig, über die die Applikation ein Plug-in notifizieren muss, damit der Ansatz funktioniert. Viele dieser Ereignisse können jedoch auch mit Hilfe von Funktionalitäten des Betriebssystems erfasst werden. So lässt sich beispielsweise jeder Kopiervorgang applikationsunabhängig anhand des Windows-Clipboards erkennen. Das Öffnen oder Schließen einer Applikation und auch Speichervorgänge an Dokumenten, soweit ihnen eine Änderung des Dokuments vorausging - lassen sich mit Hilfe der Programmierschnittstelle des Windows-Betriebsystems feststellen. Das Einfügen von wiederverwendeten Elementen kann, falls es über Tastaturkürzel ("STRG+V") erfolgt, mit Hilfe eines Tastatur-Hooks erkannt werden. Auch Informationen über das momentan geöffnete Dokument können in den meisten Fällen mit Hilfe des Fenstertitels und Funktionen des Betriebssystems ermittelt werden. Einzig das Einfügen kopierter Inhalte, wenn dies mit Hilfe der Benutzeroberfläche der Applikation erfolgt, wie zum Beispiel in Form eines Einfügenknopfes oder Menu-Eintrags, muss durch die API der Applikation als Ereignis zur Verfügung gestellt werden. Weiterhin ist ein Zugriff auf die Inhalte des Dokumentes selbst, also Text und Objekte, notwendig, um Relationen validieren zu können. Auch dies sollte durch eine API gegeben sein. Es ist also möglich, den größten Teil der notwendigen Informationen und Ereignisse unabhängig von der Applikation zu erhalten. Da dies jedoch in vielen Fällen mit erhöhtem Aufwand und gesteigerter Fehleranfälligkeit verbunden ist, sollten wenn vorhanden, Ereignisse und Informationen, die die applikationsspezifische API bietet, verwendet werden.

Das vom PowerPoint-Plug-in verwendete Konzept der vollständigen Nachverfolgung hat weitaus höhere Anforderungen an die gegebene Programmierschnittstelle. So müssen alle in Kapitel 7.2.2 und 4.1 aufgeführten Aktionen, die dazu führen können, dass eine Relation ihre Validität verliert, von der API an das Plug-in weitergegeben werden. Auch an das Format des Dokumentes an sich sind die Anforderungen höher. Zudem funktioniert eine Nachverfolgung nur bei stark strukturierten Dokumentformaten.

Da der Filesystem-Monitor auf Betriebsystemebene arbeitet ist hier implizit eine Übertragbarkeit auf die meisten Windows-Systeme gegeben.

7.3 Verwaltung

Nachdem Beziehungsinformationen durch die Plug-ins erfasst wurden, werden sie im lokalen Cache zwischengespeichert, bevor sie auf den zentralen Server übertragen werden. Im Folgenden wird zunächst

die grundlegende Art und Weise der Verwaltung der Informationen über Dokumente und erfasste Relationen beschrieben, bevor die zugehörigen Komponenten auf Client- als auch auf Server-Seite erläutert werden.

7.3.1 Grundlegende Aspekte

Ein Dokument kann durch Vervielfältigung, zum Beispiel in Form von Sicherheitskopien oder durch Übertragung per Mail, über eine Vielzahl von Instanzen verfügen. Diese können bearbeitet und angepasst und wiederum vervielfältigt werden. Das LIS.KOM-System soll Lebenszyklusinformationen im Hintergrund erfassen ohne die Nutzung und Bearbeitung von Dokumente durch Nutzer zu beeinträchtigen. Eine Versionierung, wie sie zum Beispiel SVN- oder CVS-Systeme [Gru86] bieten, ist nicht das Ziel. Daher werden lediglich physikalisch existierende Instanzen von Dokumenten betrachtet und keine virtuellen Versionen oder Revisionen. Weiterhin wird, wie in Kapitel 2.3.2 beschrieben, jede physikalische Instanz als eigenständiges Dokument behandelt. Verwandtschaften und Identität von Dokumentinstanzen werden mit Hilfe von Variantenrelationen ausgedrückt (siehe Kapitel 2.4.3).

ID-Vergabe und Konfliktbehandlung

Ein wichtiger Aspekt der Verwaltung von Lebenszyklusinformationen ist die Zuordnung der gesammelten Lebenszyklusinformationen zu dem entsprechenden Dokument. Eine solche Zuordnung muss über ein einzigartiges, unveränderliches Merkmal des Dokuments erfolgen. Keins der gegebenen Merkmale eines Dokuments, wie zum Beispiel URL, Titel oder Dateiname und keine der existierenden Metadaten können hierfür verwendet werden. Der absolute Pfad einer Datei ist zwar auf einem bestimmten System einzigartig, kann jedoch auf einem anderen System derselbe sein. Außerdem kann sich dieser im Laufe des Lebenszyklus eines Dokuments ändern. Auch die existierenden Metadaten einer Datei, die vom Betriebssystem zur Verfügung gestellt werden, enthalten kein eindeutiges Merkmal. Deshalb wird jedes Dokument, wie im in Kapitel 4.2.3 beschriebenen Schema vorgesehen, mit einer einzigartigen ID versehen. Dadurch kann ein Lebenszyklusinformationssatz eindeutig einem bestimmten Dokument zugeordnet werden. Das Windows-Betriebssystem sieht für die meisten Dokumenttypen die Möglichkeit vor, eine Datei mit eigenen Metadaten zu versehen. Hier wird die ID des Dokuments gespeichert. Eine solche ID ist zwar durch den Nutzer nicht unveränderbar - theoretisch kann ein Nutzer die Metadaten einer Datei öffnen und die ID verändern - im normalen Gebrauch kann dies jedoch ausgeschlossen werden. Wenn die ID in den Metadaten eines Dokuments gespeichert wird, hat das zur Folge, dass sie beim Kopieren, Verschieben oder Versenden per Mail erhalten bleibt. Dies hat zwei Auswirkungen, die bei der Verwaltung der Daten im LIS.KOM-Framework berücksichtigt werden müssen: Die Entstehung von lokalen oder globalen Konflikten als negative Auswirkung, sowie die Nachvollziehbarkeit von Beziehungen zwischen Versendeten oder anderweitig zwischen verschiedenen Systemen transportierten Dokumenten als positive Konsequenz.

Wird ein Dokument vervielfältigt, enthält die Kopie dieselbe ID wie das Quelldokument. Die ID ist dann nicht mehr eindeutig. Dies muss jedoch innerhalb des jeweils betrachteten Systems der Fall sein, da sonst ein Konflikt entsteht und zu einem Dokument erfasste Lebenszyklusinformationen nicht mehr

eindeutig zugeordnet werden können. Es gibt zwei verschiedene Arten von Konflikten: lokale Konflikte und globale Konflikte. Erstere entstehen, wenn auf einem lokalen System, beispielsweise dem Rechner eines Nutzers, ein Dokument mit ID vervielfältigt wird. Geschieht dies zur Laufzeit des Plug-ins der jeweiligen Applikation kann der Kopie des Dokuments direkt eine neue ID zugewiesen und somit die Entstehung eines Konflikts verhindert werden. Andernfalls muss die Vergabe einer neuen ID entweder durch eine im Hintergrund laufende Applikation erfolgen - zum Beispiel den FileSystem-Monitor (siehe oben) - oder es wird im Nachhinein mittels einer Heuristik entschieden, ob und wann eine neue ID vergeben werden muss. Diese Heuristik untersucht bei Zugriff auf ein Dokument, ob dessen ID schon im lokalen Cache vorhanden ist. Daraufhin wird anhand verschiedener Parameter wie Pfad, Computer- und Benutzername entschieden, ob es sich bei dem Dokument um eine Kopie handelt, oder nicht. Wenn nicht, werden falls nötig die zu dem Dokument erfassten Daten im lokalen Cache aktualisiert. Handelt es sich bei dem Dokument jedoch um eine Kopie - ist also ein Konflikt aufgetreten - so wird dem Dokument eine neue ID zugewiesen und eine Variantenbeziehung zwischen den beiden Dokumentinstanzen erfasst. Die Heuristik ist im Folgenden als Pseudocode dargestellt. Sie überprüft, ob ein geöffnetes Dokument identisch mit den zu dieser ID gespeicherten Informationen im lokalen Cache ist. Wenn die Methode *falsch* zurückliefert, muss eine Variantenbeziehung erzeugt werden, wenn sie *wahr* zurückliefert, müssen gegebenenfalls die im lokalen Cache gespeicherten Informationen mit dem Zustand des Dokumentes selbst synchronisiert werden, falls sich zum Beispiel der Pfad oder Dateiname geändert hat.

Algorithm 7.3.1: HEURISTIK ZUR AUFLÖSUNG LOKALER KONFLIKTE(LG)

Hole OfficeDoc-Objekt aus dem lokalen Cache

Erhalte OfficeDoc-Objekt aus geöffnetem Dokument

if $Comptername1 == Computername2$

then
- **if** $Pfad1 == Pfad2$
  - **then return** $(wahr)$
  - **else**
    - **if** $DateiExistiert(Pfad1)$ && $DateiExistiert(Pfad2)$
      - **then return** $(falsch)$
      - **else return** $(wahr)$

else
- **if** $Besitzer1 == Besitzer2$ && $Pfad1 == Pfad2$
  - **then return** $(wahr)$
  - **else return** $(falsch)$

Ein globaler Konflikt entsteht, wenn ein Dokument von einem Nutzer zu einem anderen übertragen wird. Dieser Konflikt kann erst aufgelöst werden, wenn eine Synchronisation beider lokalen Caches mit dem LIS.KOM-Server stattgefunden hat. Vor der Synchronisierung des lokalen Caches mit dem LIS.KOM-Server werden grundsätzlich alle lokalen Konflikte aufgelöst. Das Verfahren zur Auflösung eines globalen Konfliktes läuft dabei ähnlich ab, wie bei lokalen Konflikten: Eine Heuristik entscheidet auf Grund von verschiedenen Parametern, ob ein Konflikt besteht und löst ihn auf, indem eine neue ID vergeben und eine Variantenrelation gezogen wird.

Dadurch, dass die ID eines Dokuments bei einer Übertragung erhalten bleibt, kann vorteilhafterweise eine Relation zwischen den Dokumenten bei Sender und Empfänger, unabhängig von der Art und Weise der Übertragung und ohne Einsatz von Plug-ins in den zum Versenden verwendeten Applikationen erfasst werden. Wenn ein Dokument versendet wird, kann die entstehende Variantenbeziehung zwischen den Dokumentinstanzen auf Sender und Empfängerseite anhand der Identität der IDs nachvollzogen werden. Dieser positive Nebeneffekt ermöglicht eine Übertragung und Vervielfältigung von Dokumenten unter verschiedenen Nutzern und auf verschiedenen Systemen ohne Einschränkungen.

Eine Variantenrelation entsteht immer dann, wenn ein Dokument vervielfältigt wird. Es kann also angenommen werden, dass alle Element-, Aggregations- und Verknüpfungsrelationen des Ursprungsdokuments auch für die Variante gelten. Deshalb werden bei der Entstehung einer neuen Variantenrelation alle dem Urpsrungsdokument zugeordneten Relationen der drei genannten Typen ebenso der Variante zugewiesen. Die Relationen werden hierbei dupliziert und die im Schema vorgesehenen Felder für Zielbeziehungsweise Quelldokument mit den Informationen des neuen Dokuments versehen. Dadurch ergibt sich eine Vererbung bzw. Ableitung von Relationen entlang des Variantenstrangs. Eine Möglichkeit wäre auch, die Relationen nicht zu duplizieren sondern sie lediglich anhand der Variantenbeziehung, also über das Ursprungsdokument der Variantenbeziehung als Mittler, dem neuen Dokument zuzuordnen. In einem solchen Fall gehen jedoch Informationen verloren, wenn im Nachhinein Relationen des Ursprungsdokuments entfernt werden, die in der Variante noch enthalten sind.

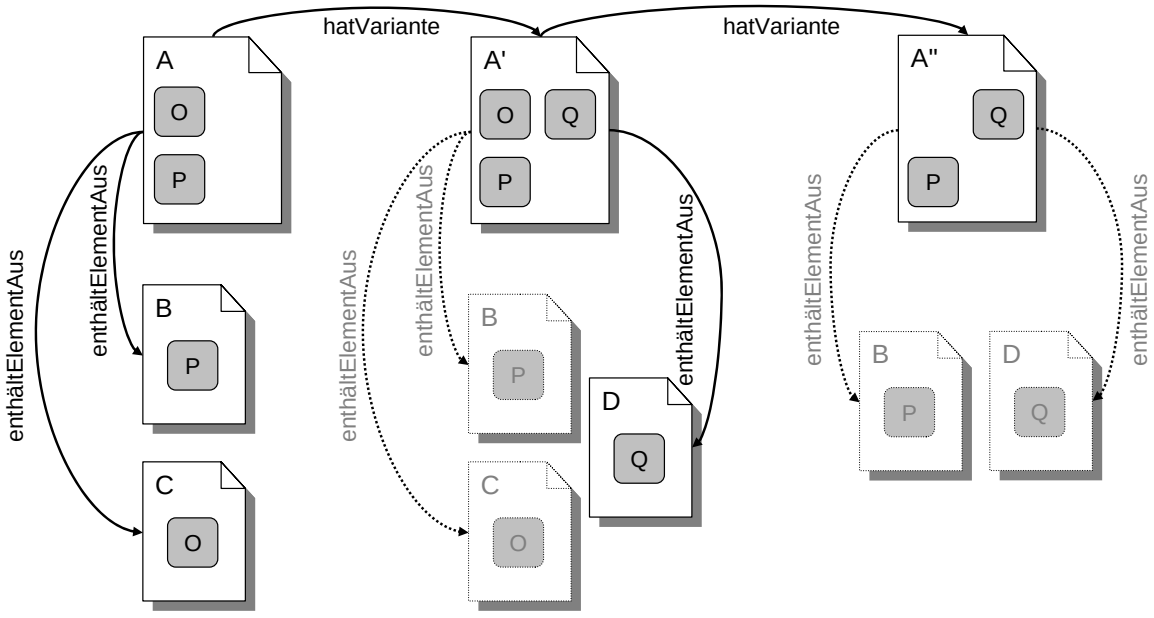

Abbildung 7.4.: Vererbung von Relationen

Abbildung 7.4 zeigt die Vererbung von Relationen entlang eines Variantenstrangs. Die Inversen zu den gegebenen Relationen wurden lediglich der Übersichtlichkeit halber weggelassen. Für diese gelten die Ausführungen im Folgenden jedoch ebenso. Dokument A' erbt die beiden Elementrelationen von Dokument A zu Dokument B und C, hat jedoch durch Bearbeitung eine weitere Relation zu Dokument D

erhalten. Dokument A" erbt wiederum die Relationen von Dokument A', wurde jedoch so verändert, dass die Relation zu Dokument C ihre Validität verloren hat und deshalb bei der Validierung entfernt wurde.

Datenmodell

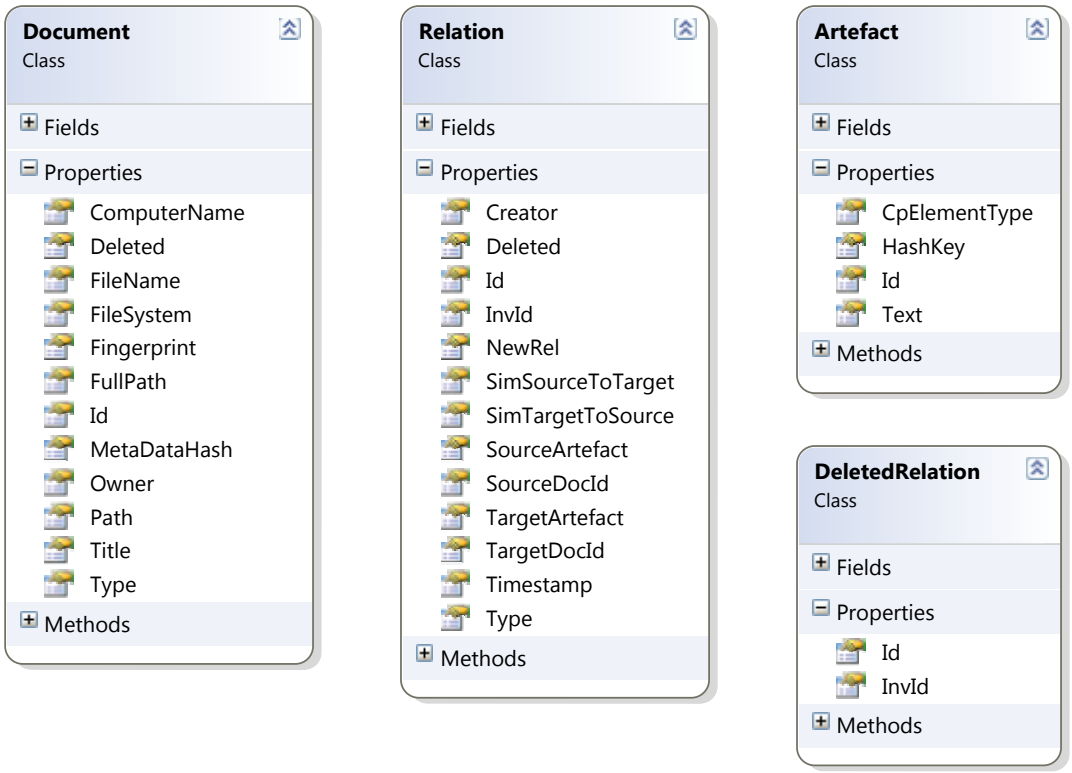

Abbildung 7.5.: Objektmodell des LIS.KOM Backends

Das konzipierte Schema zur Verwaltung von Lebenszyklusinformationen wurde wie beschrieben umgesetzt. Im Objektmodell des LIS.KOM-Frameworks (siehe Abbildung 7.5) gibt es demnach jeweils Klassen für Dokumente, Relationen sowie für diesen untergeordnete Artefakte. Außerdem ist es notwendig, die IDs gelöschter Relationen ebenso vorzuhalten (siehe 7.3.2). Informationen über den Speicherort wurden in das Dokument-Objekt integriert. Im Backend - also der Datenbank, in der die Informationen vorgehalten werden - werden die Artefakte in derselben Tabelle wie zugehörigen Relationen gespeichert. Es gibt demnach zwei Haupttabellen zur Verwaltung der Informationen - eine für die Dokumente und eine für die Relationen.

7.3.2 LIS.KOM-Client und LIS.KOM-Server

Es gibt zwei Komponenten im LIS.KOM-Framework, die für die Verwaltung der Lebenszyklusinformationen zuständig sind. Das ist zum einen der LIS.KOM-Client und zum anderen der zentrale LIS.KOM-Server.

Der LIS.KOM-Client ist die clientseitige Komponente des LIS.KOM-Frameworks zur Verwaltung der erfassten Informationen. Er besteht hauptsächlich aus dem lokalen Cache, in dem die auf dem System des Nutzers gesammelten Daten zwischengespeichert werden. Dieser ist in Form einer MYSQL-Datenbank [WA02] unter Verwendung des NHibernate-Frameworks [NHi09] umgesetzt. Der LIS.KOM-Client bietet eine Schnittstelle an, mit deren Hilfe die Erfassungs-Plug-ins die Daten in die lokale Datenbank übertragen und sie von dort abrufen können. Zu gegebenen Zeitpunkten - immer wenn ein überwachtes

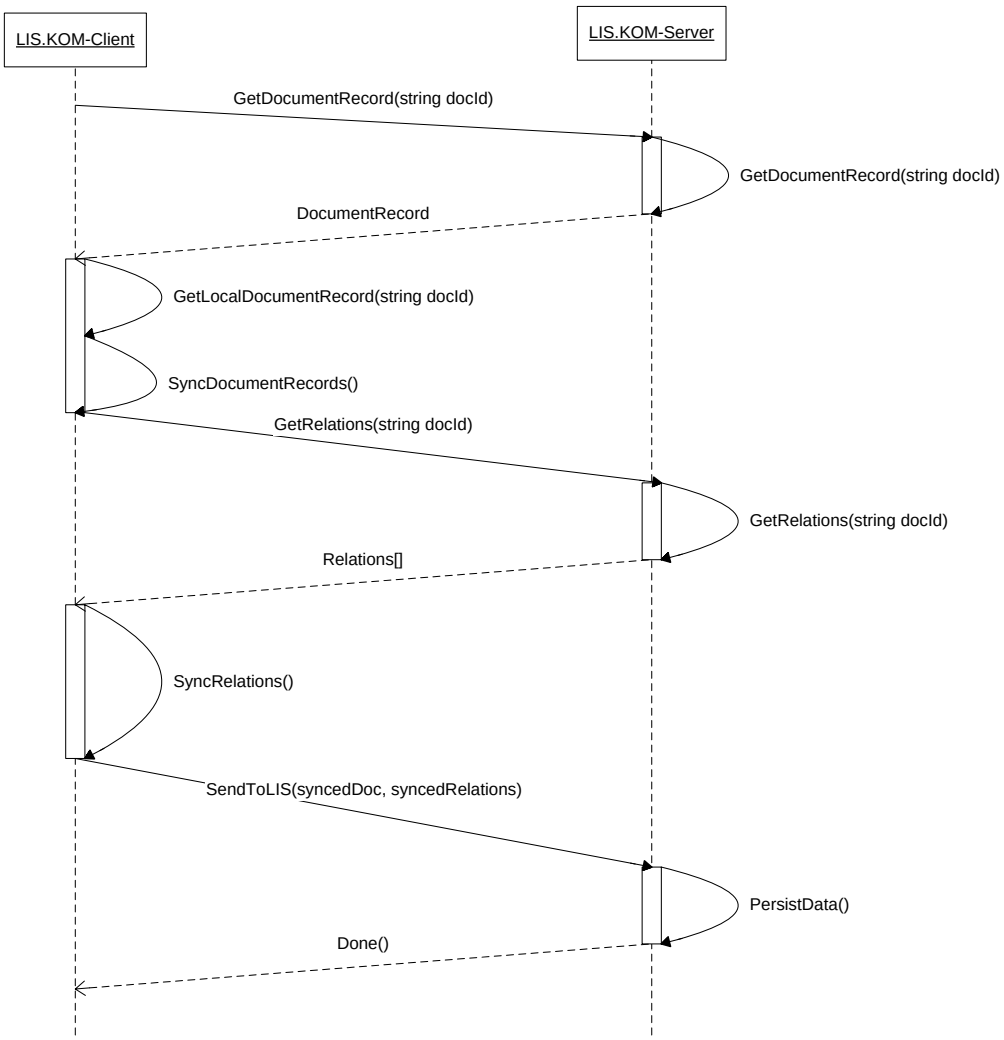

Abbildung 7.6.: Synchronisierung der Lebenszyklusinformationen eines Dokuments

Dokument beziehungsweise eine überwachte Applikation geschlossen wird - werden die zu lokalen Dokumenten gesammelten Informationen mit dem zentralen LIS.KOM-Server synchronisiert. Dieser Prozess kann je nach Menge der Dokumente und Relationen vergleichsweise viel Zeit in Anspruch nehmen und damit das Schließen eines Dokuments erheblich verzögern. Damit sich dies nicht störend auf die Arbeit eines Nutzers auswirkt, ist die Synchronisierung in eine eigene Applikation und somit einen eigenen Prozess ausgelagert, der von den Plug-ins gestartet wird. Beim Schließen eines einzelnen Dokuments werden die Informationen zu diesem Dokument sowie dessen Relationen synchronisiert. Beim Schließen der gesamten Applikation, werden alle lokalen Dokumente und deren Relationen synchronisiert. Zusätz-

lich besteht zu diesem Zeitpunkt die Möglichkeit, die Binärdaten der Dokumente selbst in des Repository auf dem LIS.KOM-Server zu übertragen. Im Folgenden wird die Synchronisierung eines einzelnen Dokuments anhand eines Sequenzdiagramms erläutert (Abbildung 7.6).

Wie beschrieben wird vorausgesetzt, dass die im lokalen Cache gespeicherten Daten konsistent mit dem tatsächlichen Zustand auf dem System des Nutzers sind. Daraufhin werden zunächst die für das Dokument selbst erfassten Daten - also der Dokument-Knoten des Lebenszyklusinformationsschemas - abgeglichen. Ist das Dokument beziehungsweise dessen ID nicht auf dem LIS.KOM-Server vorhanden, kann es dort einfach angelegt werden, ohne Konflikte zu verursachen. Im Fall eines Konflikts wird dieser nach der oben beschriebenen Methode aufgelöst und gegebenenfalls eine neue Variantenrelation angelegt. Hierbei wird die ID des Dokuments auf Client-Seite im Dokument und im lokalen Cache angepasst. Die synchronisierten Informationen werden aus Performance-Gründen nicht sofort an den Server übertragen, sondern gemeinsam mit allen anderen Daten am Schluss des Synchronisierungsprozesses.

Nun werden alle Relationen zu dem Dokument vom Server abgerufen und im Einzelnen mit den lokal zu dem Dokument vorhandenen Relationen verglichen. Wenn eine Relation auf der einen Seite vorhanden ist, auf der anderen jedoch nicht, gibt es zwei Möglichkeiten: Entweder die Relation wurde auf der einen Seite gelöscht, ist aber auf der anderen Seite noch vorhanden, oder die Relation ist auf der einen Seite neu und muss auf der anderen Seite hinzugefügt werden. Anhand der Relation selbst kann nicht entschieden werden, wie zu verfahren ist. Deshalb wird auf beiden Seiten - sowohl im lokalen Cache des LIS.KOM-Clients als auch auf dem LIS.KOM-Server - eine Liste geführt, die IDs von Relationen enthält, die zwischenzeitlich gelöscht wurden. Wenn eine serverseitige Relation in der lokalen Liste gelöschter Relationen vorhanden ist, wird sie vom Server gelöscht und in die Liste gelöschter Relationen auf dem Server eingetragen und andersherum. So wird aus den beiden Mengen an Relationen auf Client und Server eine konsistente Schnittmenge gebildet und schließlich zusammen mit den synchronisierten Dokumentinformationen und Informationen über gelöschte Relationen an den Server übertragen.

Wenn alle lokalen Dokumente und ihre Relationen synchronisiert werden sollen, wird der beschriebene Vorgang für jedes Dokument ausgeführt. Die Übertragung der Daten finden auch hier nur einmal am Ende des Prozesses statt. Diese Art der Synchronisierung bewirkt, dass nur jeweils die Informationen zu lokal vorhandenen Dokumenten sowie alle Relationen zu diesen Dokumenten jedoch nicht zwingend auch die Informationen zu allen Quell- beziehungsweise Zieldokumenten abgeglichen werden. Wenn ein lokales Dokument eine Variantenrelation zu einem Dokument eines anderen Nutzers besitzt, wird zwar die Relation, nicht jedoch die Informationen zu dem anderen Dokument selbst in den lokalen Cache übertragen. Es kommt also vor, dass es Relationen gibt, die auf Dokumente zeigen, die nicht in der lokalen Datenbank vorhanden sind - wohl aber auf dem LIS.KOM-Server. Für eine konsistente Erfassung und Verwaltung der Informationen spielt dies keine Rolle. Lediglich bei der Nutzung der Informationen muss dieser Aspekt berücksichtigt werden.

Eine Übertragung der Binärdaten in das Repository auf dem LIS.KOM-Server ist ebenso vorgesehen. Die Dokumente werden anhand ihrer ID im Filesystem auf dem Server abgelegt. Bei der Übertragung eines Dokuments von einem Client zum Server, wird zunächst mit Hilfe eines MD5-Hashwertes überprüft, ob das Dokument bereits vorhanden ist und ob es sich geändert hat. Wenn es noch nicht vorhanden ist oder sich geändert hat, wird das Dokument übertragen und eine eventuell vorhandene Version durch die neue ersetzt. Die Verwendung des Repositories ist optional, da in einigen Szenarien, insbesondere

in Unternehmen Dokumente oftmals vertraulich sind und nicht ohne weiteres dupliziert werden dürfen. In einem Open-Content-Szenario (wie zum Beispiel MIT OpenCourseWare [Lon02]) kann ein solches Repository jedoch sehr hilfreich für die Nutzung der Informationen sein, da gewünschte Dokumente stets verfügbar gemacht werden können, auch wenn sie auf den Rechnern anderen Nutzer abgespeichert sind.

7.4 Nutzung

Für die Nutzung der durch das LIS.KOM-Framework gewonnenen Informationen gibt es, wie in Kapitel 4.3 beschrieben, eine Vielzahl von Möglichkeiten. Der Fokus dieser Arbeit liegt jedoch auf der Erfassung sowie der Verwaltung von Lebenszyklusinformationen. Zur Vervollständigung des die ganze Arbeit motivierenden Szenarios, wurden zwei unterschiedliche Arten der Nutzung prototypisch implementiert: Einerseits kann die Bearbeitung von Dokumenten durch Nutzungs-Plug-ins, die direkt in die jeweilige Applikation integriert sind, unterstützt werden. Andererseits wird die Verwaltung von Dokumenten mit Hilfe eines erweiterten Explorers für das Dateisystem unterstützt.

7.4.1 Nutzungs-Plug-ins

Nutzungs-Plug-ins wurden prototypisch für PowerPoint und Word umgesetzt. Im wesentlichen werden durch diese Plug-ins zwei Hauptfunktionalitäten unterstützt: Anzeige von und Zugriff auf verwandte Dokumente und Registrierung von Notifikationen. Das Nutzungs-Plug-in ermöglicht es dem Nutzer sich schnell und ohne den Arbeitskontext zu wechseln, Dokumente anzeigen zu lassen, die mit dem geöffneten Dokument in Beziehung stehen. Dies ist besonders dann hilfreich, wenn Relationen über mehrere bestehende Verbindungen geschlossen werden. Dies ist beispielsweise der Fall, wenn die Variante des geöffneten Dokuments, die ein anderer Nutzer bearbeitet hat, Elemente eines weiteren Dokuments dieses Nutzers verwendet, die auch relevant sein könnten. Die Dokumente können wiederum direkt in der Applikation geöffnet und bearbeitet werden. Eine weitere Möglichkeit der Nutzung stellt das Registrieren von Notifikationen dar. Oftmals wissen Autoren nicht wie und wo ihre Dokumente von welchen Nutzern wiederverwendet werden. Die Plug-ins ermöglichen es, sowohl für das geöffnete Dokument als auch für mit diesem in Beziehung stehende Dokumente Notifikationen zu registrieren. Es gibt zwei mögliche Ereignisse, die notifiziert werden können: Die Änderung eines verwandten Dokuments (jedoch nicht des geöffneten Dokuments selbst) sowie die Erzeugung von Relationen zu einem Dokument. Dadurch wird der Nutzer beim Öffnen des Dokument notifiziert, wenn ein verwandtes Dokument geändert wurde, beziehungsweise wenn das gerade geöffnete oder ein verwandtes Dokument eine neue Relation erhalten haben. In einem solchen Fall erscheint eine Nachricht, die dem Nutzer das Ereignis mitteilt. Zusätzlich können erfolgte Notifikationen auch in einem über einen Tabulator erreichbaren Teil der Oberfläche des Plug-ins betrachtet werden. Bei Notifikationen werden alle Relationen bis zu einer konfigurierbaren Strukturtiefe in Betracht gezogen. Abbildung 7.7 zeigt die Oberfläche des Nutzungs-Plug-ins für PowerPoint. In dem gewählten Reiter werden die Elementrelationen gemeinsam mit einer Vorschau auf die entsprechenden Artefakte angezeigt. Das Kontextmenu ermöglicht das Öffnen des Dokuments sowie die Registrierung von Notifikationen für die zwei genannten Ereignisse. Für eine Notifikation wird jeweils

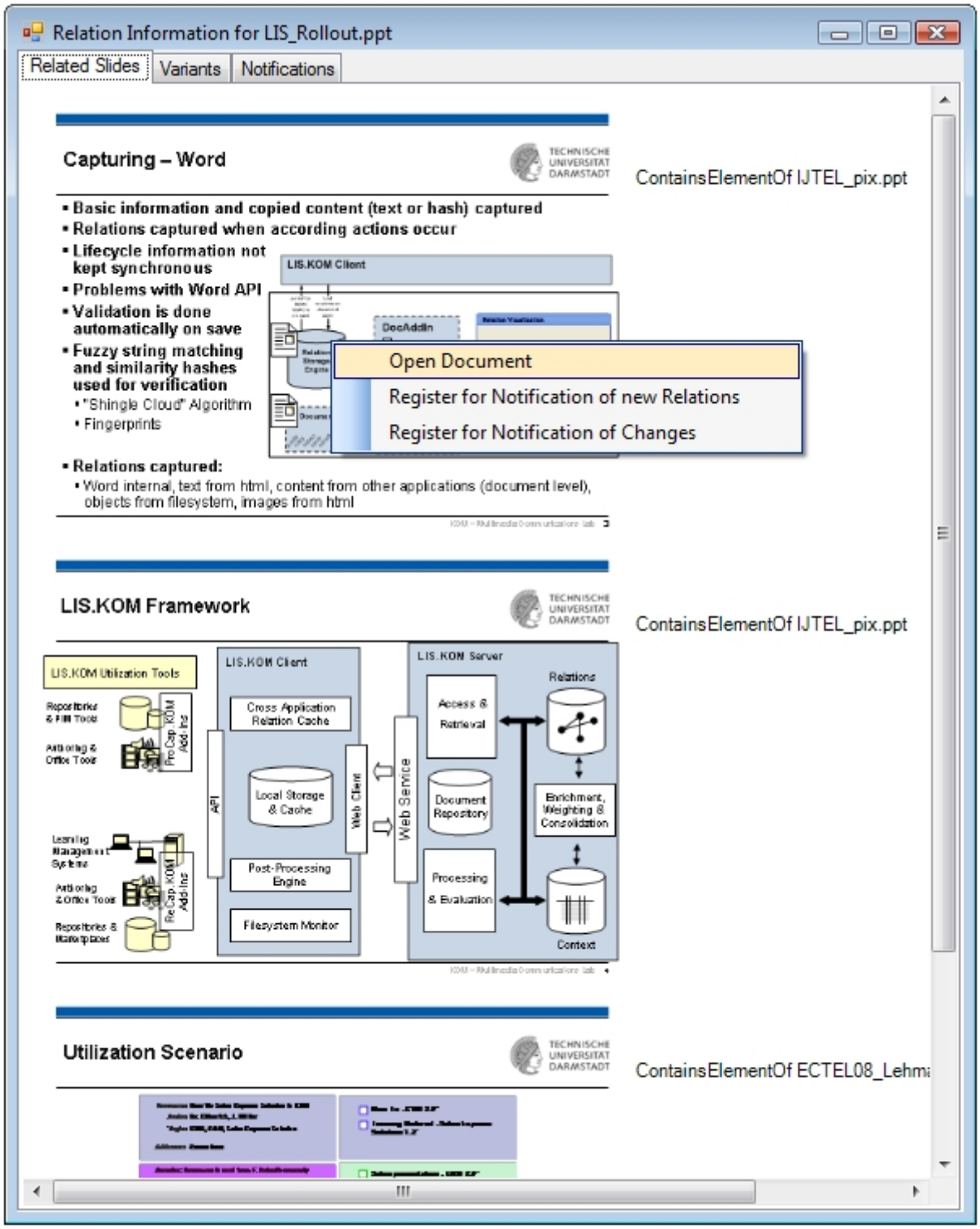

Abbildung 7.7.: Das Nutzungs-Plug-in für PowerPoint

der Zeitstempel der Notifikation - für die Notifikation auf neue Relationen - beziehungsweise ein MD5-Hashwert des Dokumentes - für die Notifikation bei Änderungen - in einer lokalen Datenbank gespeichert. Beim Öffnen eines Dokuments werden die Daten der für dieses Dokument erfassten Notifikationen mit den Daten des LIS.KOM-Servers verglichen und entsprechend Notifikationen generiert. Wenn statt des MD5-Hashwerts ein MiLe-Fingerprint oder ein anderer ähnlichkeitssensitiver Hashwert verwendet wird, ist es sogar möglich einen Schwellwert an Änderung anzugeben, der überschritten werden muss, bevor eine Notifikation erfolgt.

7.4.2 LIS.KOM-Explorer

Der LIS.KOM-Explorer ist eine standalone Applikation zur Nutzung von Lebenszyklusinformationen. Die zu Grunde liegende Idee ist es, den bestehenden Windows-Explorer, den die meisten Windows-Nutzer zur Verwaltung des Dateisystems verwenden, zu erweitern und mit Beziehungsinformationen anzureichern. Da die Erweiterbarkeit des Windows-Explorers jedoch nicht in der notwendigen Art und Weise gegeben ist, wurde ein bestehender Klon des Windows-Explorers verwendet, dessen Quellcode frei verfügbar ist [Roe09]. Da er sich fast genauso bedienen lässt wie der original Windows-Explorer, kann er als sinnvolle Basis für ein Proof-of-Concept genutzt werden. Der LIS.KOM-Explorer wurde um einen Bereich zur kontextsensitiven Anzeige von Beziehungsinformationen zu dem jeweils gewählten Dokument erweitert. Dadurch werden zu jedem gewählten Dokument in Beziehung stehende weitere Dokumente angezeigt. Es werden Funktionen zum Browsen des Ordners, in dem das Dokument abgelegt ist oder zum direkten Öffnen des Dokuments in der ihm zugewiesenen Applikation bereitgestellt. Abbildung 7.9 zeigt die Gesamtansicht des LIS.KOM-Explorers. Die linken beiden Bereiche gehören zu dem erwähnten Explorer-Klon und dienen dem Browsen des Dateisystems, wie im original Windows-Explorer. Der rechte Bereich stellt die Erweiterung dar. Hier werden die Relationen des im mittleren Bereich ausgewählten Dokuments sowie verschiedene Zusatzinformationen dargestellt. Die Relationen sind in einem Baum nach Relationstyp angeordnet und können nach Bedarf ein- oder ausgeklappt werden. Über ein Kontextmenu können die entsprechenden Dokumente geöffnet oder in die übergeordneten Ordner gewechselt werden. So kann sich ein Nutzer an den Beziehungen entlanghangeln, und es ist möglich auch Dokumente, die in höhergradiger Beziehung zu dem gewählten Dokument stehen, aufzufinden.

Abbildung 7.8 zeigt in einem Ausschnitt die Darstellung der Beziehungsinformationen. Die Beziehungen werden auf Dokumentebene dargestellt. Wenn es mehrere Relationen zwischen demselben Quell- und Zieldokument gibt, werden diese zusammengefasst (konsolidiert). Die gegebene Anzahl gibt an, wieviele Relationen zwischen dem gewählten Dokument und dem im Baum angezeigten Zieldokument bestehen. Außerdem werden der Zeitpunkt der Erstellung der Relation, der Autor des Zieldokuments sowie der Erzeuger der Relation und der Rechner, auf dem das Dokument zu finden ist dargestellt. Wenn das Dokument sich auf einem fremden Rechner befindet, wird beim Öffnen zunächst versucht das Dokument aus dem Repository des LIS.KOM-Server herunterzuladen. Ist es dort nicht verfügbar, kann der Besitzer kontaktiert werden, um das gewünschte Dokument zu erhalten.

In den weiteren Registern können die einzelnen Relationstypen getrennt dargestellt und gegebenenfalls visualisiert werden. Dafür bietet sich eine Visualisierung in Form eines Graphen an, womit die Beziehungen eines Dokuments dargestellt und verfolgt werden können.

Es sind noch verschiedene weitere Erweiterungen der Funktionalität des LIS.KOM-Explorers denkbar. So könnte zum Beispiel eine Suchfunktion hinzugefügt werden, die, wie in Kapitel 4.3 beschrieben, die Beziehungen eines Dokuments in das Ranking des Suchergebnisses einbezieht. Dokumente, die viele Beziehungen besitzen, können höher gerankt werden als Dokumente ohne Beziehungen. Auch der Typ der Relationen kann unterschiedlich gewichtet werden. Variantenrelationen können höher gewertet werden als Elementrelationen oder Aggregationsrelationen. Schließlich kann auch ein Vergleich der Autoren der in Beziehung stehenden Dokumente in Betracht gezogen werden. Die Wiederverwendung eines Dokuments oder Artefakts durch einen fremden Autor ist möglicherweise höher zu gewichten als eine durch den Autoren des Quelldokuments.

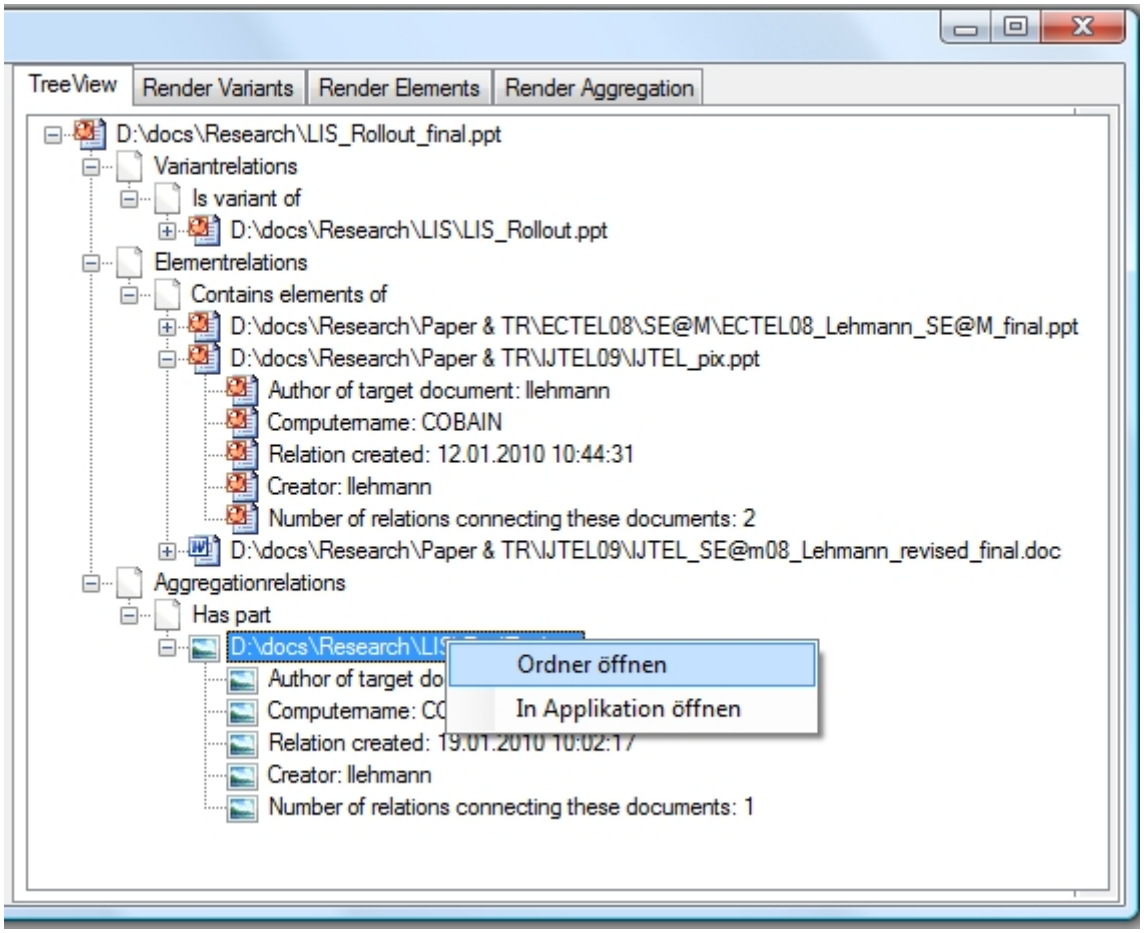

Abbildung 7.8.: Nahansicht der Beziehungsinformationen im LIS.KOM-Explorer

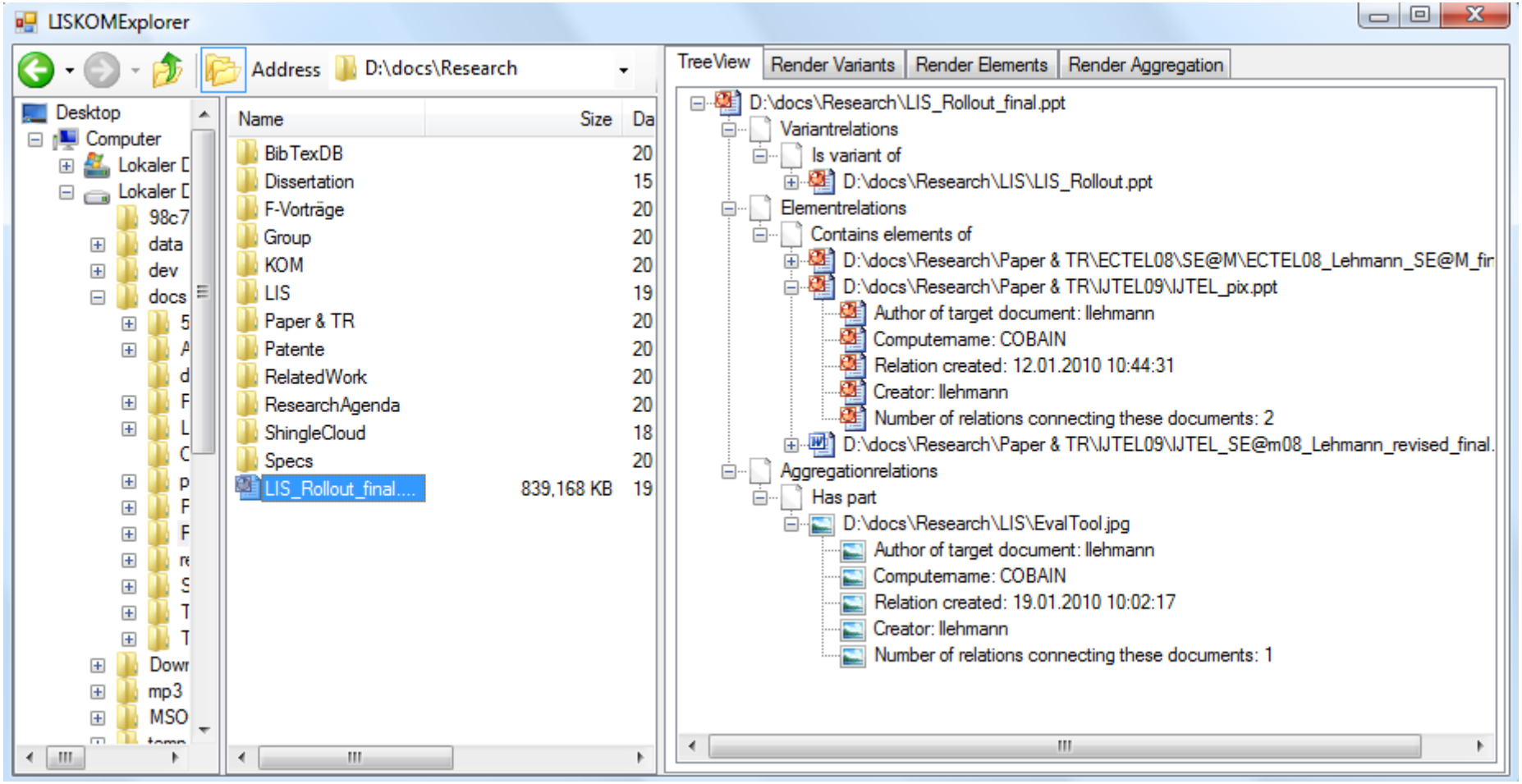

Abbildung 7.9.: Gesamtansicht des LIS.KOM-Explorers

8 Evaluation des LIS.KOM-Frameworks

In diesem Kapitel erfolgt die Evaluation der in Kapitel 7 beschriebenen Implementierung des LIS.KOM-Frameworks hinsichtlich der Erfassung von Beziehungsinformationen. Zunächst werden die Methodologie der Auswertung sowie deren Voraussetzungen erläutert, bevor die Ergebnisse sowohl in quantitativer als auch qualitativer Hinsicht dargestellt werden.

8.1 Vorgehen bei der Evaluation der Erfassung von Beziehungsinformationen

8.1.1 Evaluationsszenario

Es wurde die Erfassung von Beziehungsinformationen sowohl in Microsoft PowerPoint als auch Microsoft Word nutzerbasiert evaluiert. Insgesamt nahmen 27 Personen an der Evaluation teil. Von den Teilnehmern nutzen 19 sowohl das PowerPoint- als auch das Word-Plug-in. Vier Teilnehmer nutzen nur jeweils eins der Plug-ins. Folglich gab es für jedes Plug-in insgesamt 23 Nutzer. Gründe dafür, dass nicht bei allen Teilnehmern beide Plug-ins installiert wurden, waren vor allem Gegebenheiten des betreffenden Systems oder individuelle Wünsche der Teilnehmer. Bei allen Teilnehmern handelt es sich um wissenschaftliche Mitarbeiter eines Lehrstuhls, also um Wissensarbeiter, deren Arbeitsabläufe abwechslungsreich und dynamisch sind. Die Erstellung von PowerPoint- oder Word-Dokumenten ist ein Prozess, den alle Teilnehmer schon des Öfteren durchgeführt haben und dies auch in regelmäßigen Abständen immer wieder tun. Zusätzlich zu den Plug-ins wurden auf dem System jedes Nutzers noch folgende Applikationen installiert:

1. Der Filesystem-Monitor zur Überwachung von relevanten Ereignissen, die außerhalb von Word und PowerPoint auftreten (zum Beispiel das Kopieren, Verschieden oder Umbenennen von Dateien), als dritte Erfassungskomponente,
2. Eine MySQL-Datenbank als lokaler Cache zum Zwischenspeichern der erfassten Informationen,
3. Die SyncApp, durch welche die Synchronisation des lokalen Caches mit dem zentralen LIS.KOM-Server aus den Plug-ins ausgelagert wird.

Die Plug-ins wirken sich nicht auf die Benutzeroberfläche der Applikationen aus, sind dem Nutzer auf den ersten Blick also verborgen. Über ein Menu der jeweiligen Applikation ist es jedoch möglich die installierten Plug-ins anzuzeigen und gegebenenfalls zu deaktivieren. Ein Ziel bei der Realisierung der Plug-ins war, dass sie sich nicht auf die Benutzbarkeit der jeweiligen Applikation auswirken. Dieses Ziel wurde beim PowerPoint-Plug-in erreicht. Das Word-Plug-in wirkt sich aufgrund des durch die Validierung der Informationen notwendigen Rechenaufwands etwas auf die Benutzbarkeit der Applikation aus, da einige dedizierte Vorgänge merklich mehr Zeit in Anspruch nehmen, wenn das Erfassungs-Plug-in installiert ist. Hierzu zählt zum Beispiel das Speichern eines Dokuments. Dies behindert ein Arbeiten mit der Applikation jedoch nicht in signifikanter Art und Weise. Es kann deshalb bei beiden Applikationen

davon ausgegangen werden, dass das Vorhandensein der Plug-ins die Nutzer nicht in ihrer Arbeitsweise mit den Applikationen beeinflusst hat. Der Hawthorne-Effekt [Ada84], der bewirkt, dass sich das Wissen von Probanden um ihre Teilnahme an einer Evaluation auf deren Ergebnisse auswirken kann, kann in diesem Fall folglich vernachlässigt werden.

Für die Evaluation wurden die Ergebnisse aus einer Laufzeit des Systems von acht Wochen verwendet. Die Teilnehmer haben in dieser Zeit die relevanten Applikationen wie sonst auch verwendet. Sie wurden weder zur Benutzung der Applikationen noch zur Wiederverwendung von Inhalten in irgendeiner Form ermutigt. Die Nutzer hatten während der Evaluationsphase die Möglichkeit, die Plug-ins auf Wunsch zu deaktivieren, zum Beispiel bei der Bearbeitung von vertraulichen Dokumenten. Dadurch bestand die Möglichkeit, dass ein Nutzer Änderungen an einem Dokument vornahm, die nicht von den Plug-ins erfasst werden konnten. Weiterhin war das Evaluationsszenario nicht geschlossen. Dabei können auch Nutzer die Dokumente bearbeiten, die nicht Teilnehmer der Evaluation sind, auf deren System demnach auch keine Plug-ins installiert sind. Dadurch kann es passieren, dass einige Relationen ihre Validität verlieren. Dies ist insbesondere für PowerPoint-Dokumente ein Problem, da hier eine komplette Nachverfolgung der Aktionen auf dem Dokument notwendig ist, um die Validität der Relationen gewährleisten zu können. Innerhalb des PowerPoint-Plug-ins erfolgte demnach keine Validierung der erfassten Relationen zur Laufzeit, sondern die Relationen wurden wie in Kapitel 4 beschrieben nachverfolgt.

Um die Auswirkungen des offenen Szenarios etwas abzumildern, wurde zusätzlich im Nachhinein eine naive Form der Validierung von PowerPoint-Relationen eingesetzt. Alle Element- und Aggregationsrelationen in PowerPoint besitzen ein Artefakt, welches die ID der PowerPoint-Folie enthält, auf der die kopierten Inhalte eingefügt wurden. Wenn diese Folie in der entsprechenden Präsentation nicht mehr vorhanden ist, weil sie zum Beispiel in einer nicht überwachten Umgebung gelöscht wurde, wird diese Relation entfernt. Die innerhalb des Word-Plug-ins erfassten Aggregations- und Elementrelationen wurden - wie beschrieben - zu regelmäßigen Zeitpunkten zur Laufzeit des Systems validiert.

8.1.2 Zielsetzung und Methodologie

Das primäre Ziel der Evaluation ist es, zu zeigen, dass eine Erfassung von Beziehungsinformationen aus dem Lebenszyklus von Wissensdokumenten mit hoher Verlässlichkeit durchführbar ist. Es wurde demnach evaluiert, ob die vom LIS.KOM-Framework während der Evaluation erfassten Relationen gültig sind. Dies wird im weiteren Verlauf des Kapitels als "qualitative Evaluation" bezeichnet. Das sekundäre Ziel ist eine quantitative Erhebung der erfassten Relationen. Hierbei geht es einerseits darum, die Annahme, die in der Motivation getroffen wurde, dass Wiederverwendung ein wichtiger Aspekt im Lebenszyklus von Wissensdokumenten ist, zu bestätigen. Andererseits soll erhoben werden in welchem Umfang Wiederverwendung erfolgt, welche Relationstypen am häufigsten auftreten und wie das Wiederverwendungsverhalten der Benutzer im Vergleich ist. Auch die quantitative Entwicklung der Relationen über einen längeren Zeitraum ist interessant, unter anderem für eine Dimensionierung des LIS.KOM-Systems. Für die quantitative Evaluation wurden die während der achtwöchigen Laufzeit erfassten Relationen auf ihre Verteilung bezüglich Relations-, Dokumenttypen und Teilnehmern hin untersucht.

Die nach acht Wochen Laufzeit erhobenen Daten wurden vor der Auswertung grundlegend gesäubert, d.h. korrupte Relationen und nicht in Relationen genutzte Dokumentdaten aus der Datenbank entfernt.

Korrupte Relationen umfassen Relationen, die kein Inverses oder keine ID besitzen oder deren Quell- und Zieldokument identisch sind. Solche Relationen sind im LIS.KOM-System prinzipiell von der Erfassung ausgeschlossen, können jedoch in bestimmten Situationen, wie bei Applikationsfehlern oder vereinzelten Abstürzen auftreten. Zu jedem geöffneten Dokument wird generell ein Datensatz angelegt. Es wird aber nicht notwendigerweise auch eine Relation mit diesem Dokument erfasst, daher werden Dokumentensätze, die an keiner Relation beteiligt sind, vor der Auswertung aus der Datenbank entfernt.

Um die qualitative Evaluation durchführen zu können, müssen die Relationen manuell beurteilt werden. Dazu muss zum Zeitpunkt der Beurteilung einer Relation Zugriff auf die Binärdaten von Quell- und Zieldokument der Relation bestehen, da nur unter Berücksichtigung des Inhalts der Dokumente beurteilt werden kann, ob eine erfasste Relation gültig ist, oder nicht. Deshalb wurden alle an Relationen beteiligten Dokumente der Teilnehmer zu bestimmten Zeitpunkten in das auf dem LIS.KOM-Server befindliche Repository übertragen. Es gibt Fälle, in denen dieser Prozess nicht korrekt durchgeführt wurde, so dass nicht für alle Relationen die Quell- und Zieldokumente im Repository verfügbar sind. Dies tritt beispielsweise auf, wenn ein Dokument lokal gelöscht wird, bevor die Binärdaten ins Repository übertragen werden können. Weitere Gründe für eine gescheiterte Übertragung sind Auslastung des Servers, die Größe der Dokumente oder nicht Vorhandensein einer Internetverbindung zum Übertragungszeitpunkt. Demnach sind nicht alle erfassten Relationen bewertbar oder im Nachhinein validierbar. Für die qualitative Evaluation können nur die Relationen berücksichtigt werden, bei denen sowohl Quell- als auch Zieldokument im Repository vorhanden sind. Dies gilt auch für eine im Nachhinein durchzuführende Validierung. Eine Ausnahme ergibt sich bei Webseiten: Wenn Inhalte von Webseiten wiederverwendet werden, werden die Webseiten selbst nicht ins Repository übertragen. Demnach gilt hier auch die genannte Voraussetzung nicht.

Da die in Abschnitt 8.1.1 beschriebene nachgelagerte Validierung der PowerPoint-Relationen auch für eine quantitative Auswertung notwendig ist, da sonst die Zahlen durch ungültige Relationen verfälscht werden, können auch für die quantitative Auswertung nur die Relationen, deren Binärdaten im Repository vorhanden sind, berücksichtigt werden.

Evaluiert werden die im LIS.KOM-Framework implementierten Relationstypen - also Aggregationsrelationen, Elementrelationen und Variantenrelationen. Bei der qualitativen Evaluation wird manuell anhand der Quell- und Zieldokumente und gegebenenfalls der Quell- und Zielartefakte beurteilt, ob eine Relation gültig oder ungültig ist. Hierbei werden verschiedene Gültigkeitskategorien unterschieden (siehe Abschnitt 8.1.3). Für die Unterstützung der manuellen Bewertung der Relationen wurde ein Werkzeug entwickelt, das in Abschnitt 8.1.4 näher erläutert wird.

Die Relationen wurden von zwei Personen unabhängig voneinander bewertet. Dabei wurden jeweils repräsentative Teilmengen der Gesamtmenge an Relationen manuell bewertet. Die Reliabilität der Ergebnisse wird dann mit Hilfe der Kappa-Statistik bestimmt (siehe Abschnitt 8.3). Die Teilmengen wurden dabei mit Hilfe der ID, die jede Relation besitzt, ermittelt. Hierbei wurde wie bei dem in Kapitel 6.1 beschriebenen 0-mod-p Ansatz verfahren.

Abbildung 8.1 fasst das beschriebene Vorgehen bei der Evaluation des LIS.KOM-Frameworks noch einmal in Form eines Prozessmodells zusammen.

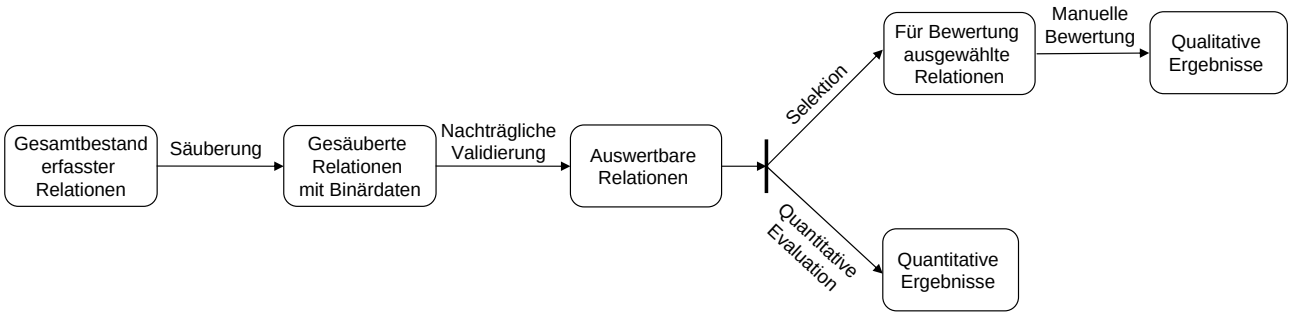

Abbildung 8.1.: Vorgehen bei der Evaluation des LIS.KOM-Frameworks

8.1.3 Gültigkeitskriterien für die qualitative Evaluation

Anstatt eine binäre Beurteilung der Relationen in gültige oder ungültige Relationen vorzunehmen, wurden für die Einordnung der Relationen bezüglich ihrer Gültigkeit mehrere Gültigkeitskategorien verwendet. Eine Einteilung der Relationen in solche ist sinnvoll, da damit eine exaktere Aussage über das System und eine Vorhersage über die Nutzbarkeit der Informationen getroffen werden kann. Zudem werden durch eine feingradigere Abstufung Grenzfälle vermieden. Folgende Stufen der Validität werden unterschieden:

- **Gültig**: Die Relation ist so wie sie erfasst wurde gültig. Für *Aggregationsrelationen* bedeutet dies, dass das Quelldokument erkennbar im Zieldokument, an der durch das Zielartefakt gegebenen Stelle enthalten ist. Bei *Elementrelationen* gilt dasselbe, nur muss auch die im Quellartefakt gegebene Position des Elements stimmen. Zudem muss das kopierte Artefakteine inhaltliche Aussagekraft und nicht lediglich strukturellen Wert besitzen. Dies zu beurteilen liegt bei der Person, die die manuelle Auswertung vornimmt. *Variantenrelationen* sind dann valide, wenn sie entweder erkennbar einem Versionsstrang eines Dokuments entstammen, oder eine große inhaltliche Ähnlichkeit zwischen den Dokumenten vorhanden ist. Eine Übersetzung stellt zum Beispiel eine valide Variante dar, genau wie eine durch Aktualisierung leicht veränderte Version eines Dokuments.

- **Vorlage** (Template): In einem solchen Fall wird ein Dokument oder Artefakt nicht aufgrund seines Inhalts, sondern aufgrund des Layouts oder der Struktur wiederverwendet. Es besteht also keine bzw. kaum inhaltliche Verwandtschaft zwischen Quell- und Zieldokument, beziehungsweise Quell- und Zielartefakt. Bei *Variantenrelationen* ergibt sich diese Einstufung wenn ein Dokument klar nachvollziehbar aus formalen bzw. strukturellen Gründen wiederverwendet wurde. Das ist zum Beispiel bei der Wiederverwendung von Formatvorlagen der Fall. Es kann aber auch der Fall sein, dass Quell- und Zieldokument dieselbe inhaltliche Struktur aufweisen, der Inhalt selbst sich jedoch unterscheidet. Bei Elementrelationen ergibt sich diese Einstufung der Validität einer Relation, wenn ein Artefakt erkennbar aufgrund seiner Form oder Struktur wiederverwendet wurde und kein inhaltlicher Zusammenhang zwischen den Artefakten zu erkennen ist. Hier kann zum Beispiel die Form eines Diagramms oder die Struktur einer einzelnen Folie ausschlaggebend für die Wiederverwendung sein. Wenn das Artefakt selbst keinen inhaltlichen Bezug besitzt, ist die Relation ebenso als Vorlage einzustufen. Wird ein einzelnes strukturelles Objekt, wie zum Beispiel ein Pfeil

oder eine geometrische Form wiederverwendet, so ist die resultierende Relation als Vorlage einzustufen. Wird andererseits ein Logo (zum Beispiel eines wissenschaftlichen Projektes) wiederverwendet, kann durchaus von einem inhaltlichen Bezug ausgegangen werden und die Relation kann als gültig bewertet werden. Bei Aggregationsrelationen ist eine Wiederverwendung als Template nicht vorgesehen. Template-Relationen sind in den meisten Anwendungsszenarien für die Nutzung von Lebenszyklusinformationen nicht anwendbar, da hier die durch Wiederverwendung implizierte inhaltliche Verwandtschaft der Dokumente den erhofften Mehrwert bieten soll und nicht eine strukturelle Verwandtschaft [PM08]. Um gültige Relationen von Template-Relationen zumindest für Variantenrelationen trennen zu können, wurde der MiLe-Algorithmus entwickelt und erfolgreich evaluiert (siehe Kapitel 6).

- **Gültig auf Relationsebene** (Artefaktfehler): Ein Artefaktfehler liegt dann vor, wenn die in der Relation gegebene Position des Zielartefakts nicht mit dem tatsächlichen Zustand im Zieldokument übereinstimmt, das Quellartefakt jedoch erkennbar im Zieldokument enthalten ist. Außerdem muss das wiederverwendete Artefakt eine inhaltliche Relevanz besitzen. Daraus lässt sich folgern, dass die Relation an sich korrekt ist, lediglich die Position beziehungsweise ID des Zielartefakts nicht korrekt verfolgt wurde. Diese Einstufung ist lediglich für Relationen, die vom PowerPoint-Plug-in erfasst wurden, relevant, da nur hier die Nachverfolgungsmethode zur Gewährleistung der Validität eingesetzt wurde. Da das Evaluationsszenario nicht geschlossen war, und außerdem die Möglichkeit der Deaktivierung des Plug-ins durch den Nutzer bestand, konnte keine vollständige Überwachung aller Relationen gewährleistet werden. Dadurch konnten Relationen unbemerkt vom LIS.KOM-System ihre Validität verlieren. In den meisten Fällen wurden Artefakte jedoch nicht gelöscht, sondern lediglich in der Position und in bestimmten Fällen auch in der ID verändert, so dass die ursprünglich erfasste Information zu dem Objekt nicht mehr gültig ist. Diese Einstufung hat nur für Element- und Aggregationsrelationen Bedeutung. In vielen Nutzungszenarien wird die exakte Position des Artefakts im Zieldokument nicht benötigt, sondern lediglich die übergeordnete Relation zwischen den Dokumenten, weswegen eine solche Relation in diesen Fällen voll nutzbar ist. Wenn ein Artefaktfehler bei einer Elementrelation auftritt, bei der das Artefakt als Vorlage wiederverwendet wurde, wird diese als ungültig eingestuft.

- **Gültig auf Dokumentebene** Eine *Elementrelation* wird als "gültig auf Dokumentebene" eingestuft, wenn das Quellartefakt nicht im Zieldokument enthalten ist - zum Beispiel weil das Löschen des Artefakts nicht verfolgt werden konnte - Quell- und Zieldokument jedoch eine deutliche inhaltliche Verwandtschaft aufweisen. Eine solche Relation ist wiederum in allen Szenarien nutzbar, wo es lediglich auf die Beziehung auf Dokumentebene ankommt. Dazu zählen zum Beispiel das Ranking oder Recommender-Systeme oder auch das Anreichern von Suchergebnissen mit verwandten Dokumenten. Auch eine Notifikation ist auf Basis von Relationen auf Dokumentebene möglich.

- **Ungültig**: Die Relation ist nicht nachvollziehbar und Quell- und Zieldokument weisen keine inhaltliche Verwandtschaft auf. Eine solche Relation ist nicht nutzbar sondern kann bei der Nutzung sogar störend wirken und beispielsweise ein Suchergebnis verfälschen.

Tabelle 8.1 zeigt eine Zusammenfassung der aufgeführten Kriterien zur Beurteilung der Relationen.

Tabelle 8.1.: Kriterien zur Bewertung von Relationen

	Variantenrelation	**Elementrelation**	**Aggregationsrelation**
Gültig	Ein Versionsstrang oder große inhaltliche Ähnlichkeit	Quellartefakt an der durch Zielartefakt gegebenen Stelle im Zieldokument enthalten und inhaltliche Relevanz des Artefakts erkennbar	Quelldokument an der durch Zielartefakt gegebenen Stelle im Zieldokument enthalten
Vorlage	Quelldokument dient erkennbar als formelle oder strukturelle Vorlage für Zieldokument	Quellartefakt an der durch Zielartefakt gegebenen Stelle im Zieldokument enthalten und keine inhaltliche Relevanz erkennbar	-
Artefaktfehler	-	Quellartefakt im Zieldokument enthalten, jedoch nicht an der durch Zielartefakt gegebenen Stelle, inhaltliche Relevanz des Artefakts erkennbar	Quelldokument im Zieldokument enthalten, jedoch nicht an der durch Zielartefakt gegebenen Stelle
Dokumentebene	-	Quellartefakt nicht in Zieldokument enthalten. Quell- und Zieldokument besitzen inhaltliche Verwandtschaft	-
Ungültig	Variantenrelation nicht nachvollziehbar	Elementrelation nicht nachvollziehbar und keine inhaltliche Verwandtschaft der Dokumente	Quelldokument nicht in Zieldokument enthalten

8.1.4 Das Evaluationswerkzeug

Zur Unterstützung der Durchführung der Evaluation wurde ein Werkzeug entwickelt, mit dessen Hilfe die evaluierende Person die Relationen durchlaufen, prüfen und bewerten kann. Es bietet Informationen zur Relation, wie Typ, Zeitstempel oder Erzeuger und Daten von Quell- und Zieldokument sowie gegebenenfalls Quell- und Zielartefakt. Wenn Text kopiert wurde, so ist dieser in der Relation gespeichert und wird ebenfalls angezeigt (vergl. Abbildung 8.2). Die Quell und Zieldokument einer Relation können direkt vom Evaluationswerkzeug aus geöffnet werden. Bei Element- und Aggregationsrelationen werden die entsprechenden Artefakte optisch hervorgehoben, so dass eine effiziente und schnelle Bewertung erfolgen kann. Der in der Relation gespeicherte Text wird ebenfalls, mittels des ShingleCloud-Algorithmus (siehe Kapitel 5), im Dokument hervorgehoben. Bei Variantenrelationen von Word-Dokumenten werden automatisch übereinstimmende Sequenzen in den Dokumenten hervorgehoben. Nichtsdestotrotz muss eine manuelle Überprüfung der hervorgehobenen Bereiche erfolgen. Das Vorgehen ist dabei wie folgt: Wenn die Relation gültig ist, genügt es, dies anhand der hervorgehobenen Bereiche zu bestätigen. Sollte die Validität anhand dieser Bereiche nicht erkennbar sein, müssen auch die restlichen Teile des Dokuments überprüft werden, um zu entscheiden, wie die Relation einzustufen ist. Dabei werden die in Abschnitt 8.1.3 aufgeführten Kriterien angewandt.

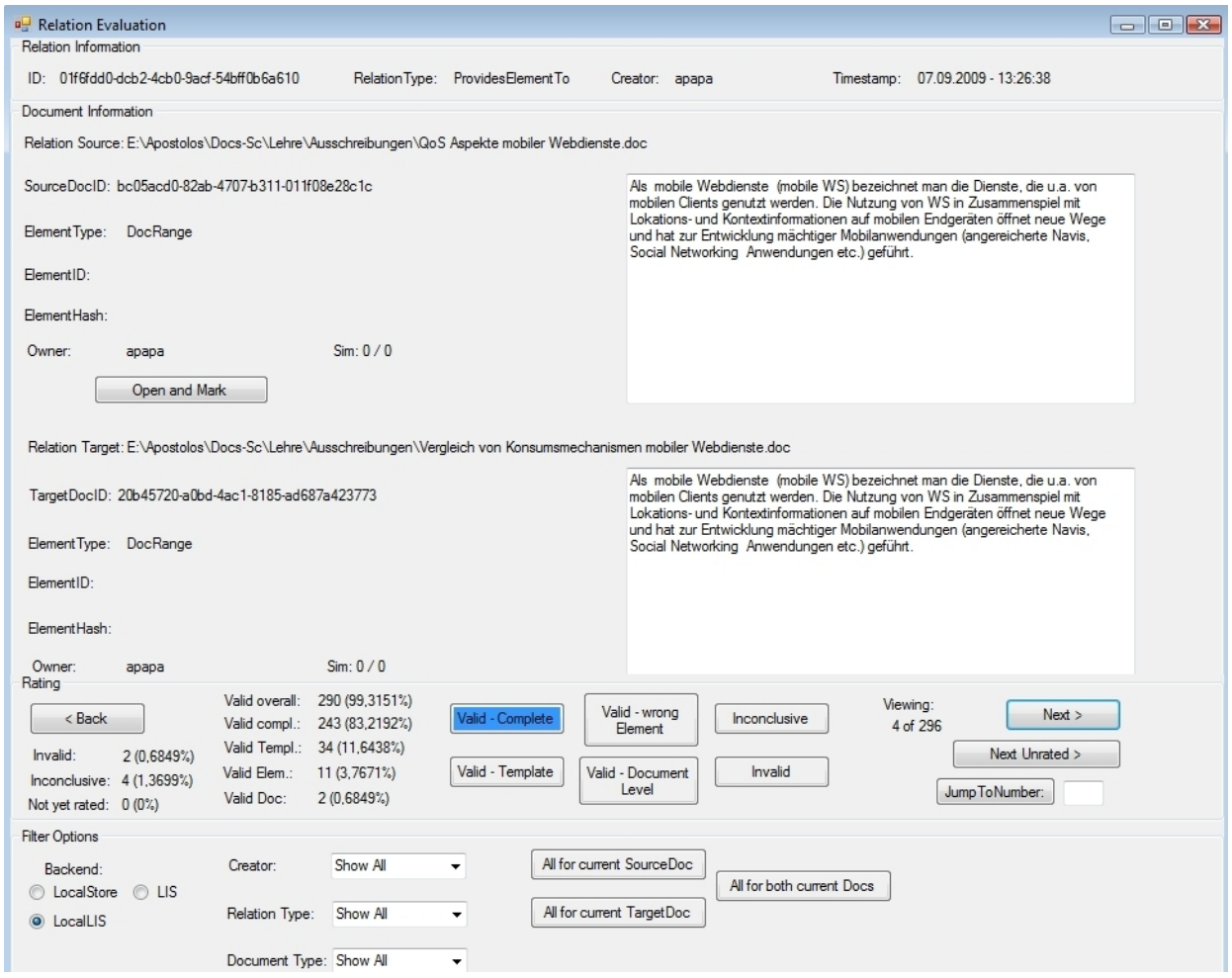

Abbildung 8.2.: Das Evaluationswerkzeug

Die Bewertung der Relation erfolgt mit Hilfe der im unteren Bereich befindlichen Buttons. Neben Funktionen zum Navigieren durch die Relationen, bietet das Werkzeug noch diverse Möglichkeiten zum Filtern, so dass die gegebenen Relationen nach Erzeuger, Zieldokumenttyp (Word oder PowerPoint) und Relationstyp ausgewählt werden können. Zudem besteht die Möglichkeit, alle Relationen, die ein bestimmtes Dokument besitzt, anzuzeigen.

Aus den bewerteten Relationen wird automatisch der Anteil der Relationen mit der jeweiligen Bewertung berechnet. Zusätzlich zu den gegebenen Gültigkeitskategorien gibt es noch die Möglichkeit, eine Relation als uneindeutig ("inconclusive") zu bewerten. Diese geht dann nicht in die Berechnung ein. Diese Möglichkeit ist dazu gedacht Relationen auszuschließen, die nicht beurteilbar sind, weil zum Beispiel Quell- oder Zieldokument nicht zugegriffen werden können.

Im Folgenden werden die Ergebnisse der Evaluation in quantitativer Hinsicht sowie die Ergebnisse der qualitativen Bewertung der Relationen präsentiert.

8.2 Quantitative Evaluation der Erfassung von Beziehungsinformationen

Bei den erfassten Relationen werden zwei Ausprägungen unterschieden: Die *Basisrelationen* sind Relationen, die aus direkten Aktionen eines Nutzers heraus entstanden sind, wie sie in den Kapiteln 4 und 7 beschrieben werden. *Abgeleitete Relationen* hingegen werden aus Relationen, die ein Dokument bereits besitzt, abgeleitet, wenn dieses vervielfältigt, also eine Variantenrelation dazu angelegt wird. Ausschließlich Aggregations- und Elementrelationen werden auf diese Weise abgeleitet. Dieses Vererben von Relationen entlang des Variantenstrangs eines Dokuments, wurde bereits in Kapitel 7.3 eingehend erläutert. Basisrelationen und abgeleitete Relationen werden in den folgenden Betrachtungen zumeist getrennt voneinander behandelt.

8.2.1 Allgemeine Ergebnisse

Im Folgenden ist unter dem Zieldokument der Relation das Dokument zu verstehen, welches als Ziel eines Kopier- beziehungsweise Speichervorgangs dient. Wenn das Zieldokument ein PowerPoint-Dokument ist, dann ist von einer PowerPoint-Relation die Rede. Wird also beispielsweise ein Artefakt in ein PowerPoint-Dokument hineinkopiert, dann ist das PowerPoint-Dokument das Zieldokument der Relation und die Relation eine PowerPoint-Relation. Für Word-Dokumente und Word-Relationen gilt Entsprechendes.

Tabelle 8.2 zeigt eine Übersicht über die insgesamt während der initialen Laufzeit von acht Wochen erfassten Relationen, die Relationen deren Dokumente verfügbar sind, sowie den Effekt der im Nachhinein auf in PowerPoint erfassten Relationen angewandten Validierung.

Über die Laufzeit von acht Wochen wurden insgesamt 7260 Relationen erfasst, davon wurden 1425 als Basisrelationen durch explizite Wiederverwendungsaktionen erzeugt, während die restlichen 5835 Relationen durch Vererbung bei der Bildung von Varianten entstanden sind. Die Basisrelationen stellen den wichtigeren Teil dar, da abgeleitete Relationen in vielen Fällen auch aus dem Variantenstrang eines Dokuments gefolgert werden können (siehe Kapitel 7.3). Den größten Anteil an der Gesamtmenge der Relationen haben Elementrelationen, also Relationen, die aus der Wiederverwendung einzelner Inhalte auf derselben Aggregationsebene (meist durch Kopieren und Einfügen) entstehen (siehe Kapitel 2.4.3).

Tabelle 8.2.: Übersicht über erfasste Relationen

	Unvalidiert - alle Relationen	Unvalidiert - mit Binärdaten	Validiert - mit Binärdaten
Gesamtzahl der Relationen	7260	6383	3854
Basisrelationen	1425	1163	1143
Abgeleitete Relationen	5835	5220	2711
Elementrelationen	5893	5317	2942
Aggregationsrelationen	581	401	247
Variantenrelationen	786	665	665
PowerPoint-Relationen	6587	5822	3293
Word-Relationen	672	560	560
Andere	1	1	1
Anzahl der Dokumente	1272	1031	1029

In PowerPoint wurde etwa die zehnfache Menge an Elementrelationen erzeugt wie in Word. Dies hängt damit zusammen, dass alle Element- und Aggregationsrelationen in Word zur Laufzeit der Erfassung validiert und gegebenenfalls aus dem System entfernt werden. Es ist jedoch auch wahrscheinlich, dass bei der Erstellung von Word-Dokumenten weniger Wiederverwendung stattfindet als bei der Erstellung von PowerPoint-Dokumenten.

Wenn die Relationen, deren Binärdokumente nicht vorhanden sind, entfernt werden, verringert sich die Gesamtzahl der Relationen um ca. 12 %. Dabei verringert sich die Zahl der Relationen in jeder Kategorie ungefähr gleich stark - mit Ausnahme von Aggregationsrelationen. Die Zahl dieser Relationen verringert sich deutlich stärker als die der anderen Relationen. Dies kann damit zusammenhängen, dass Dokumente, die mehrere Aggregationsrelationen besitzen, nicht im Repository verfügbar sind.

Wird die in Abschnitt 8.1.1 beschriebene naive Validierung auf die PowerPoint-Relationen mit vorhandenen Binärdaten angewandt, verringert sich deren Zahl erheblich. Die Gesamtzahl der Relationen verringert sich durch die Validierung um circa 40 %. Allein PowerPoint-Relationen sind betroffen, da nur hier die nachträgliche Validierung angewandt wird. Da sie nur auf Element- und Aggregationrelationen Anwendung fand, sind auch Variantenrelationen hiervon unberührt. Es ist insbesondere auffällig, dass mit überwiegender Mehrheit abgeleitete Relationen durch die Validierung aussortiert werden. Dies deckt sich mit den Erwartungen: Besonders abgeleitete Relationen entstehen oft durch Varianten, die zwischenzeitlich auf einem nicht überwachten System (siehe Kapitel 4.1) in Bearbeitung waren. Somit konnten die dort vorgenommenen Änderungen nicht nachverfolgt werden (siehe Kapitel 7.2 und 4.1).

Auf die Anzahl der Dokumente hat sich die Validierung hingegen nur in sehr geringem Umfang ausgewirkt. Zwar wurden viele Relationen als ungültig erkannt, dies hatte jedoch nur in zwei Fällen zur Folge, dass ein Dokument überhaupt keine Relationen mehr besitzt. Daraus lässt sich folgern, dass fast alle der als nicht gültig gekennzeichneten Relationen zu Dokumenten gehören, die mehrere Relationen besitzen.

Prinzipiell sind nur PowerPoint- oder Word-Dokumente als Zieldokumente möglich, da nur hierfür Plug-ins für die Erfassung von Beziehungsinformationen umgesetzt wurden. Es ist jedoch auch möglich, dass Relationen für andere Zieldokumenttypen zustande kommen. Wenn ein Dokument anderen Typs

(also zum Beispiel ein PDF-Dokument) an einer Relation mit einem Word- oder PowerPoint-Dokument beteiligt ist, erhält es eine ID und wird in das System aufgenommen. Wenn dieses nun vervielfältigt wird, erfasst der Filesystem-Monitor (siehe Kapitel 7.3.2) eine Variantenrelation. Diese Art der Erfassung ist auch für andere Dokumenttypen als Word und PowerPoint möglich. Hierdurch kam im Laufe der Evaluation eine Relation zustande ("Andere" in Tabelle 8.2), die weder als PowerPoint- noch als Word-Relation einzuordnen ist.

8.2.2 Auswertung der Basisrelationen

Die Anzahl und Verteilung der Basisrelationen lässt Rückschlüsse darauf zu, wieviele Inhalte auf welche Art und Weise von den Teilnehmern der Evaluation wiederverwendet wurden. Im Folgenden wird zunächst die Verteilung der Basisrelationen anhand des Relationstyps sowie des Zieldokumenttyps analysiert (Abbildung 8.3). Ungefähr 60 % der Basisrelationen sind PowerPoint-Relationen, dementsprechend 40 % Word-Relationen. Die Anzahl der Dokumente für Word-Relationen überwiegt die Anzahl der an PowerPoint-Relationen beteiligten Dokumente (552 zu 480 Dokumente). Zudem wurden bei Word-Dokumenten mehr Variantenrelationen erzeugt. Dagegen wurden für PowerPoint-Dokumente fast dreimal soviele Elementrelationen erfasst. Dies lässt darauf schließen, dass in Word weniger Artefakte wiederverwendet werden als in PowerPoint, ein Dokument dafür öfter vervielfältigt wird. Auffällig ist, dass in den 8 Wochen, in denen Relationen erfasst wurden, keine einzige Aggregationsrelation in Word erzeugt wurde und in PowerPoint lediglich 29. Da insgesamt deutlich mehr PowerPoint-Relationen erfasst wurden bei einer beinahe gleichen Anzahl an verschiedenen Dokumenten, kann gefolgert werden, dass im Fall von PowerPoint, die Dokumente stärker untereinander vernetzt sind, also zwischen zwei Dokumenten mehr Relationen bestehen und Artefakte kopiert wurden als im Fall von Word. Der Definition nach, verbinden Variantenrelationen grundsätzlich zwei Dokumente desselben Typs (siehe Kapitel 2.4.3). Die erhobenen Aggregationsrelationen waren, wie erwähnt, allesamt PowerPoint-Relationen und hatten in fast allen Fällen Bilder als Quelldokument. Für diese Relationstypen ist demnach eine eingehende Betrachtung der Quelldokumenttypen uninteressant. Bei Elementrelationen ist dies anders. Das LIS.KOM-Framework unterstützt die Erfassung von Elementrelationen mit einer Vielzahl von Quelldokumenttypen (siehe Kapitel 7.2.1). Abbildung 8.4 zeigt die Verteilung der unterschiedlichen Quelldokumenttypen für alle Elementrelationen der Basismenge, sowohl für PowerPoint- als auch für Word-Relationen.

Die meisten der in PowerPoint erfassten Elementrelationen haben demnach als Quelldokument ebenfalls ein PowerPoint-Dokument. Dies ist auch intuitiv der häufigste Anwendungsfall: Das Kopieren von Folien oder anderen Artefakten von einem PowerPoint-Dokument in ein weiteres. Ungefähr 10 % der PowerPoint-Relationen haben ein Word-Dokument als Quelldokument. In den meisten Fällen handelt es sich dabei um Text, der aus einem Word-Dokument in eine Präsentation kopiert wurde. Verhältnismäßig selten mit jeweils weniger als 5 % der PowerPoint-Relationen, wurden Webseiten oder andere Dokumenttypen als Quelldokumente verwendet.

Im Fall von Word macht den größten Anteil auch hier die Wiederverwendung von Artefakten aus anderen Word-Dokumenten aus. Es wurde jedoch deutlich mehr Inhalt von Webseiten kopiert (41 %) als bei PowerPoint-Relationen. Bei den anderen Dokumenttypen handelte es sich sowohl bei PowerPoint- als auch bei Word-Relationen zumeist um PDF-Dokumente.

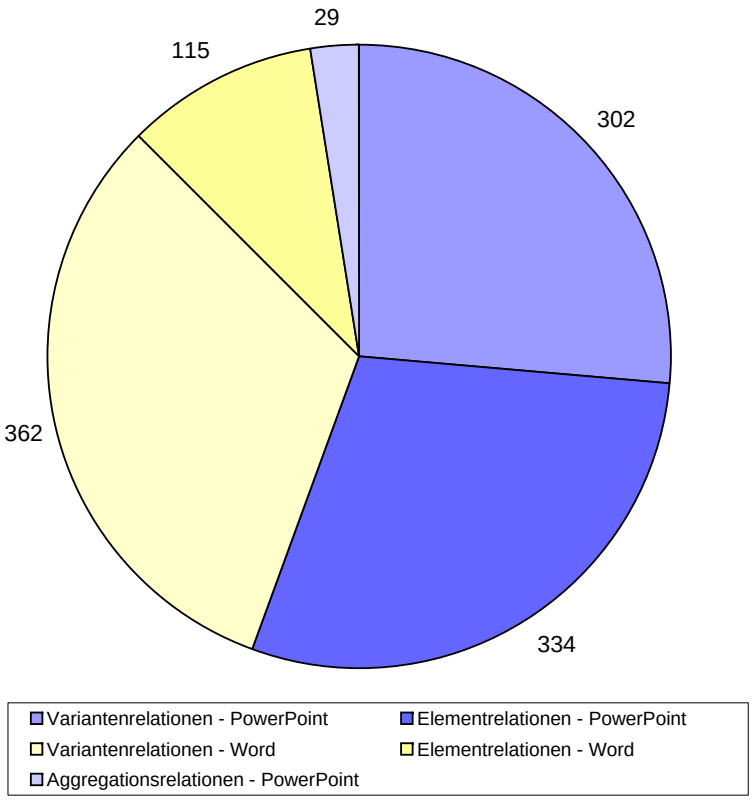

Abbildung 8.3.: Verteilung der Basisrelationen nach Relationstyp und Zieldokumenttyp

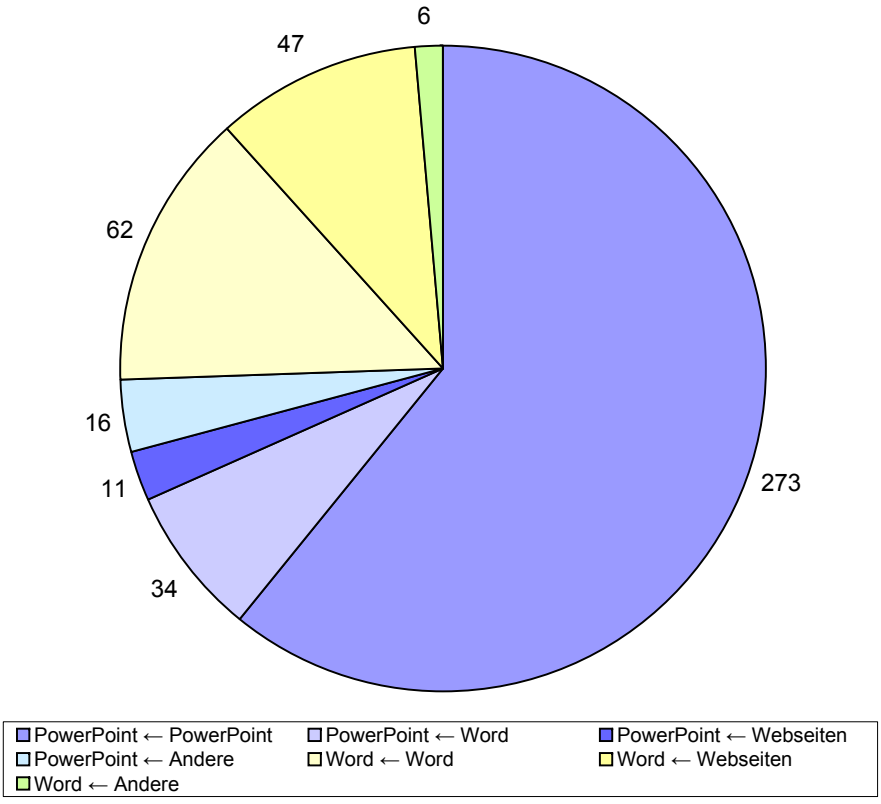

Abbildung 8.4.: Verteilung der Elementrelationen nach Ziel- und Quelldokumenttyp

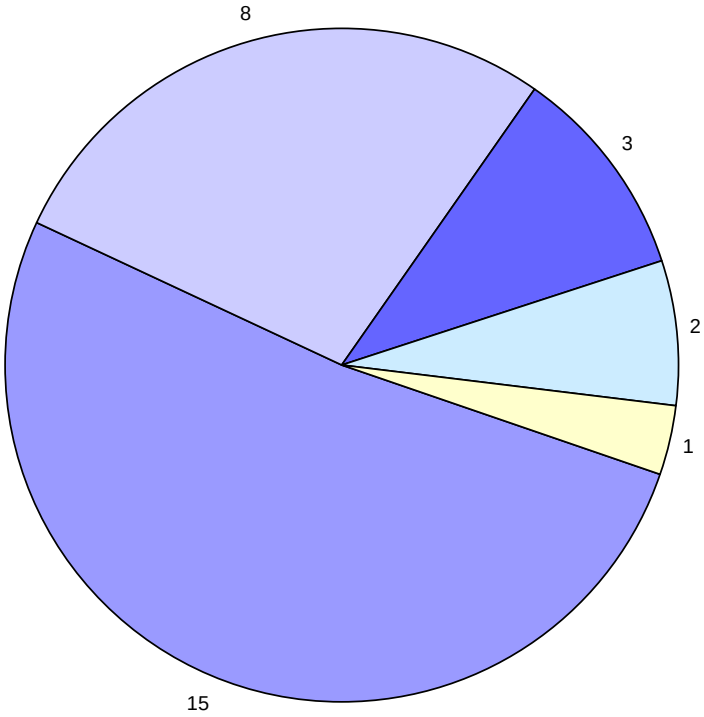

Abbildung 8.5.: Verteilung der Aggregationsrelationen auf Evaluationsteilnehmer

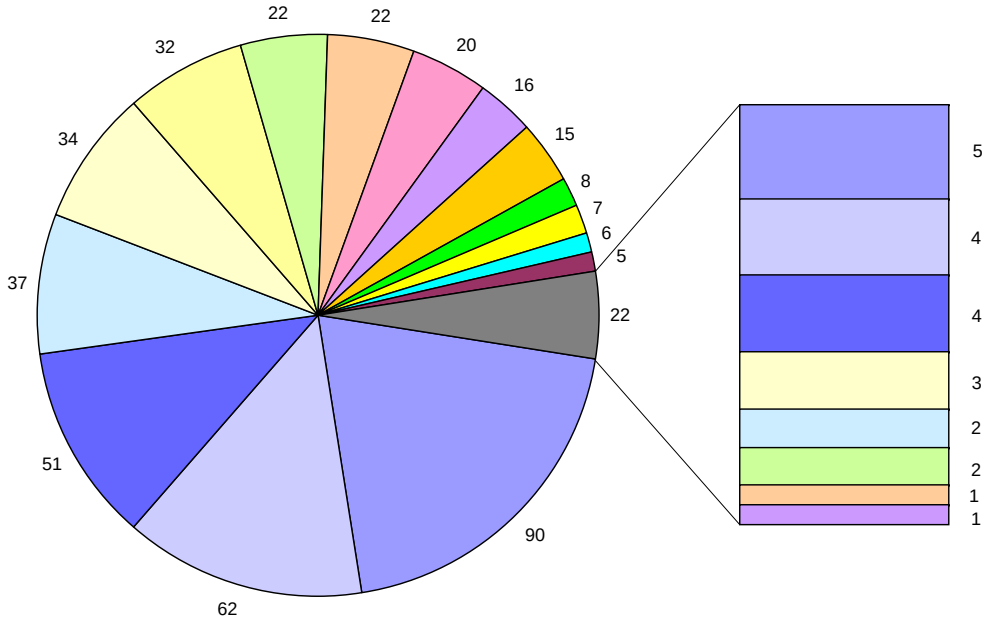

Abbildung 8.6.: Verteilung der Elementrelationen auf Evaluationsteilnehmer

Neben der Verteilung der Relationstypen nach Quell- und Zieldokumenttypen ist auch die Verteilung der erfassten Relationen auf die Teilnehmer der Evaluation interessant. Diese wird im Folgenden für Aggregationsrelationen und Elementrelationen betrachtet. Das Diagramm in Abbildung 8.5 zeigt die Verteilung der Aggregationsrelationen. Hier sind insgesamt nur 29 Relationen entstanden, die von insgesamt fünf verschiedenen Nutzern erzeugt wurden. Mehr als die Hälfte der Aggregationsrelationen wurde allein von einem Teilnehmer erzeugt. Insgesamt lässt die Verteilung und die geringe Anzahl der Relationen darauf schließen, dass Nutzer im Vergleich eher selten Wiederverwendung auf Aggregationsbasis bei Wissensdokumenten durchführen. Die geringe Anzahl und die Tatsache, dass ein einzelner Nutzer einen Großteil der Aggregationsrelationen erzeugt hat, erklärt auch die statistischen Schwankungen bei der qualitativen Evaluation für diesen Relationstyp (siehe Abschnitt 8.3).

An der Entstehung von Elementrelationen waren deutlich mehr Nutzer beteiligt. Die Anzahl der Elementrelationen der verschiedenen Nutzer im Vergleich zeigt Abbildung 8.6. Auch hier wurde ein Großteil der Relationen von wenigen Teilnehmern erzeugt: Die Hälfte der Relationen wurde allein von fünf Teilnehmern gebildet. Insgesamt haben 23 der 27 Teilnehmer Elementrelationen erstellt. Die vier Teilnehmer, die keine Elementrelationen erzeugt haben, gehören auch bei den Variantenrelationen zu den zehn Teilnehmern mit den wenigsten Relationen. Daraus lässt sich schließen, dass die überwachten Applikationen Word und PowerPoint von diesen Teilnehmern weniger intensiv genutzt wurden. Im Durchschnitt wurden pro Teilnehmer im Laufe der acht Wochen 25 Elementrelationen erzeugt. Zwölf Teilnehmer haben dabei deutlich unterdurchschnittlich viele Relationen erzeugt, acht Teilnehmer können dem Mittelfeld mit einer in der Nähe des Durchschnitt liegenden Anzahl an Relationen zugeordnet werden, und drei Teilnehmer haben deutlich überdurchschnittlich viele Relationen erzeugt. Die Unterschiede zwischen den Teilnehmern kommen auch durch die Notwendigkeit der Nutzung der entsprechenden Applikationen während der Laufzeit der Evaluation zustande: Wenn ein Teilnehmer im Zuge seiner Tätigkeit während der Laufzeit der Evaluation mehrere Präsentationen zu erstellen hatte, wurden durch ihn zwangsläufig mehr Relationen erzeugt als durch einen Teilnehmer, der während dieser Zeit überhaupt keine Präsentationen oder Word-Dokumente zu bearbeiten hatte.

8.2.3 Auswertung der abgeleiteten Relationen

Bei abgeleiteten Relationen handelt es sich ausschließlich um Aggregations- oder Elementrelationen. Sie kommen durch Vererbung zustande, wenn von einem Dokument, welches Element- oder Aggregationsrelationen besitzt, eine Variante erzeugt wird. Abbildung 8.7 zeigt die Verteilung nach Relationstyp und Zieldokument. Da es in der Basismenge keine Aggregationsrelationen mit einem Word-Dokument als Ziel gibt, können diese auch nicht vererbt werden. Mit ca. 89 % den größten Anteil an abgeleiteten Relationen haben Elementrelationen in PowerPoint. Diese hatten auch schon unter den Basisrelationen mit fast 70 % den größten Anteil bei diesen drei Relationstypen. Jede PowerPoint-Elementrelation sowie jede PowerPoint-Aggregationsrelatiom der Basismenge wurde durchschnittlich ungefähr siebenmal durch eine Variante abgeleitet. Daraus lässt sich schließen, dass im Fall von PowerPoint häufig Varianten von Dokumenten erzeugt werden, die schon mehrere Relationen besitzen. Bei Element-Relationen mit Zieldokument Word ist die Zahl hingegen rückläufig. Hier wurden nur 72 % der Relationen der Basismenge überhaupt abgeleitet. Dies hängt mit der Validierung in Word zusammen. Wenn das Zieldokument der

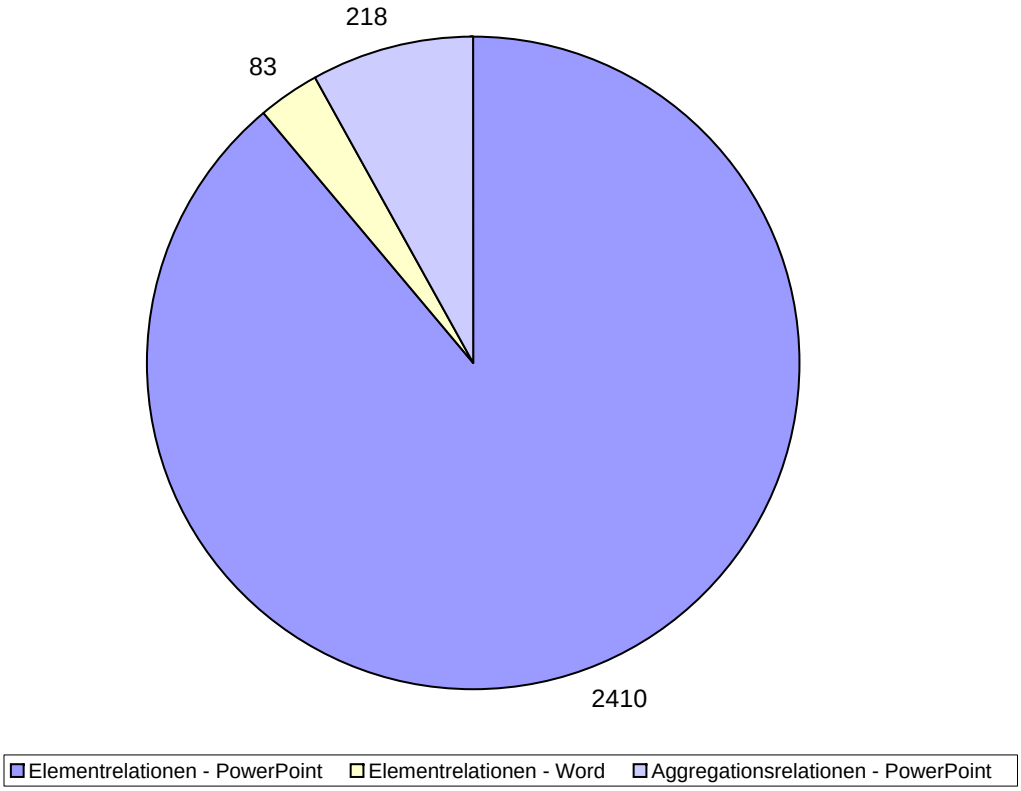

Abbildung 8.7.: Verteilung der abgeleiteten Relationen nach Relationstyp und Zieldokumenttyp

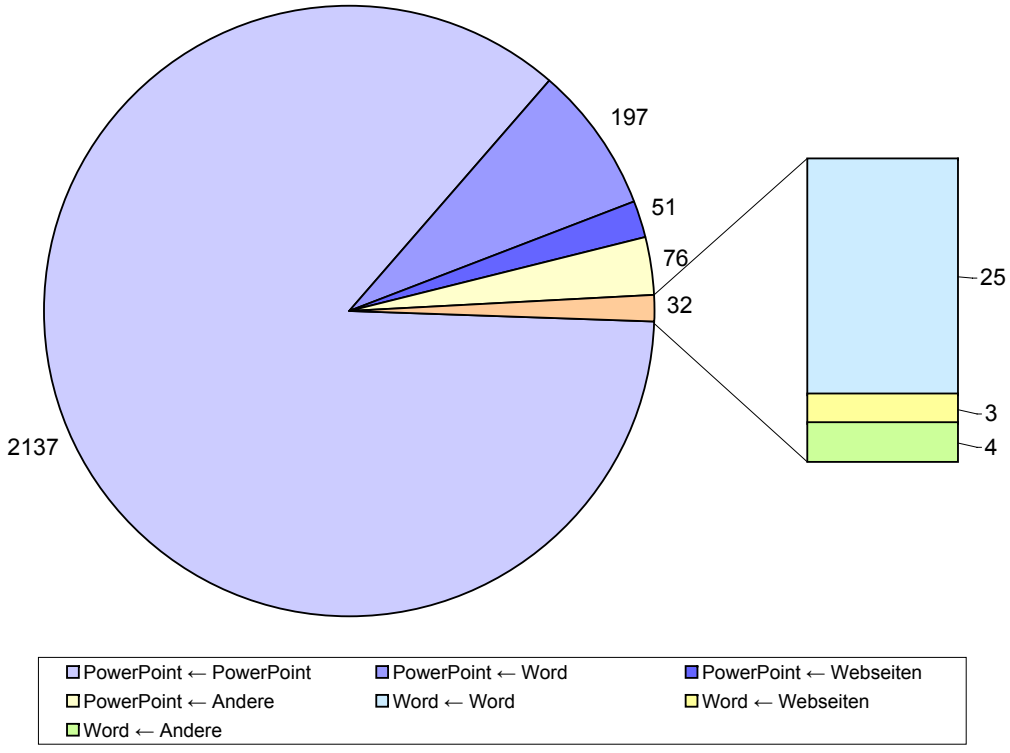

Abbildung 8.8.: Verteilung der abgeleiteten Relationen nach Ziel- und Quelldokumenttyp

Tabelle 8.3.: Quantitative Entwicklung der Relationen über den Evaluationszeitraum

	2 Wochen	4 Wochen	6 Wochen	8 Wochen	16 Wochen
Gesamtzahl aller Relationen	734	1294	2353	3854	7337
Basisrelationen	470	629	880	1143	2125
Abgeleitete Relationen	264	665	1473	2711	5212
Elementrelationen	426	896	1669	2942	5825
Aggregationsrelationen	4	15	132	247	304
Variantenrelationen	304	383	552	665	1208
PowerPoint-Relationen	447	946	1875	3293	6311
Word-Relationen	286	347	477	560	1024
Andere	1	1	1	1	2
Anzahl der Dokumente	490	619	873	1029	1811

Variantenrelation so weit verändert wird, dass die ursprüngliche Relation nicht mehr als gültig erkannt werden kann, werden die entsprechenden Elementrelationen der Variante entfernt. Ein weiterer Grund hierfür liefert die Verteilung der Quelldokumente der abgeleiteten Element-Relationen (siehe Abbildung 8.8). Bei Betrachtung der Verteilung der Quelldokumente für Word-Relationen zeigt sich, dass sich die Zahl der Relationen mit Word als Quelldokument durch Ableitung leicht gegenüber der Basismenge erhöht hat (76 gegenüber 62 Relationen). Die Zahl der Relationen mit anderen Dokumenten insbesondere aber Webseiten als Quelldokument ist jedoch um ein Vielfaches zurückgegangen. Der Grund wird bei Betrachtung der Relationen und der zugehörigen Dokumente selbst deutlich. Von einigen der Webseiten, die als Quelldokument an einer Relation beteiligt sind, wurden nur sehr kurze Ausschnitte wie URLs oder Literaturreferenzen kopiert. In einem solchen Fall hat der für die Validierung der Word-Relationen verwendete Algorithmus, so wie auch die meisten anderen Algorithmen in diesem Bereich, Probleme das Artefakt im Text des Zieldokuments aufzufinden. Wenn in der Variante der entsprechende Text leicht verändert wurde, weil zum Beispiel nur die Autoren und der Titel einer Literaturreferenz übernommen wurden, jedoch nicht die Konferenz oder der Ort, wurde das Artefakt bei der Variante nicht mehr erkannt und die Relation entfernt.

8.2.4 Quantitative Entwicklung der erfassten Relationen über den Evaluationszeitraum

Während der achtwöchigen Laufzeit wurden in regelmäßigen Zeitabständen Abbilder des Repositories und der Datenbank des LIS.KOM-Servers extrahiert. Dadurch ist es möglich die quantitative Entwicklung der Relationen über die Laufzeit der Evaluation zu betrachten. Nach der initial vorgesehenen achtwöchigen Laufzeit wurde das LIS.KOM-Framework nicht deaktiviert sondern weiterhin betrieben und somit auch der LIS.KOM-Client und die Erfassungskomponenten auf den Systemen der Teilnehmer. Dadurch konnten weiterhin Relationen erfasst werden. Daher ist es möglich, die quantitative Entwicklung der erfassten Relationen auch über die vorgesehene Laufzeit von acht Wochen hinaus zu betrachten. Tabelle 8.3 zeigt die Entwicklung der Anzahl der Relationen über die insgesamt 16-wöchige Laufzeit. Die Auswertung zeigt, dass sich die Anzahl der Basisrelationen, wie auch die Anzahl der Dokumente unge-

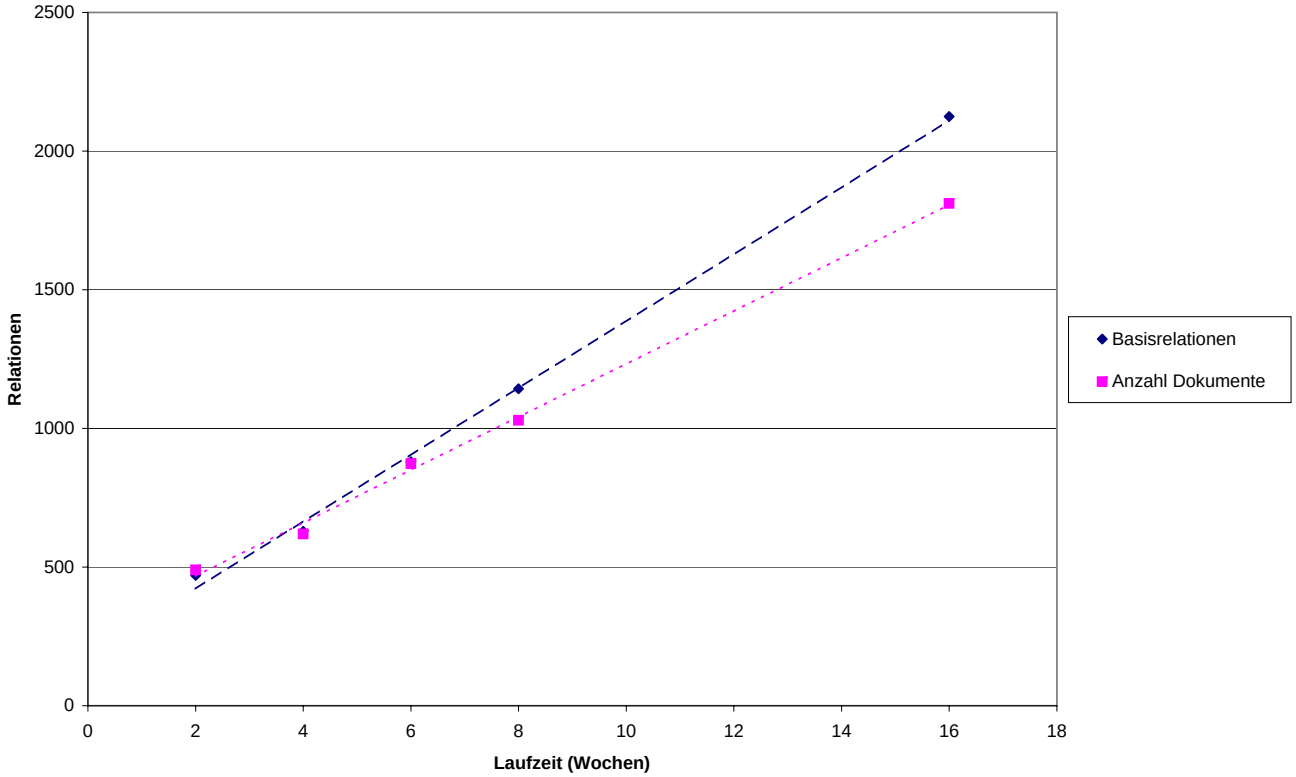

Abbildung 8.9.: Wachstum der Anzahl der Basisrelationen und Dokumente über den Evaluationszeitraum

fähr linear verhält, also nach doppelter Laufzeit ungefähr doppelt so viele Relationen bzw. Dokumente erfasst werden. Diese Annahme wird durch die in Abbildung 8.9 dargestellte Interpolation bestätigt. Hierbei wird die gemessene Menge an Relationen und Dokumenten nach zwei Wochen Laufzeit als erster Datenpunkt betrachtet, da aus organisatorischen Gründen die Erfassung nicht bei allen Teilnehmern gleichzeitig gestartet werden konnte. Dies hatte zur Folge, dass bei einigen Teilnehmern schon einige Tage vor dem geplanten Beginn der Evaluation mit der Erfassung begonnen wurde. Dadurch waren zu Beginn der Evaluation, also dem Zeitpunkt, zu dem die Teilnehmer vollständig waren, schon erfasste Daten vorhanden. Dies erklärt die konstante Verschiebung der interpolierten Geraden und somit ihren nicht im Nullpunkt liegenden Schnittpunkt mit der Y-Achse.

Die abgeleiteten Relationen lassen sich hingegen mit den vorhandenen Daten nicht aussagekräftig interpolieren. Da die Zahl der abgeleiteten Relationen sowohl der Zahl der Variantenrelationen als auch der Zahl der Aggregations- und Elementrelationen abhängt, ist ein Wachstum der Ordnung $O(n^2)$ zu erwarten. Anhand der Zahlen lässt sich zwar innerhalb der ersten acht Wochen ein Wachstum dieser Ordnung erkennen, die Kurve flacht danach jedoch deutlich ab. Dies kann beispielsweise mit der im Nachgang durchgeführten Validierung der PowerPoint-Relationen zusammenhängen.

8.3 Qualitative Evaluation der Erfassung von Beziehungsinformationen

Wie erläutert, wurden die erfassten Relationen von zwei unabhängigen Personen nach den in 8.1.2 genannten Kriterien begutachtet. Hierfür wurden repräsentative Teilmengen ausgewählt.

Tabelle 8.4.: Beurteilung der Übereinstimmung anhand von κ nach [LK77]

Kappa (κ)	**Grad der Übereinstimmung**
$\kappa < 0$	Schlechte Übereinstimmung
$0 < \kappa < 0{,}20$	Etwas Übereinstimmung
$0{,}21 < \kappa < 0{,}40$	Ausreichende Übereinstimmung
$0{,}41 < \kappa < 0{,}60$	Mittelmäßige Übereinstimmung
$0{,}61 < \kappa < 0{,}80$	Beachtliche Übereinstimmung
$0{,}81 < \kappa < 1{,}00$	(Fast) vollkommene Übereinstimmung

8.3.1 Vorgehen bei der qualitativen Evaluation

Aus den erfassten Relationen ergeben sich 1141 begutachtbare Basisrelationen, für welche die Binärdokumente verfügbar sind. Die verbleibenden zwei Relationen der Basismenge wurden im Zuge der Evaluation als nicht beurteilbar eingestuft. In beiden Fällen handelte es sich beim Quelldokument der Relation um eine durch Login geschützte Webseite. Diese konnte zum Zeitpunkt der Beurteilung nicht zugegriffen werden. Person A beurteilte die gesamte Menge der Basisrelationen, während durch Person B nur eine Teilmenge der Basisrelationen beurteilt wurde. Für die Teilmenge gilt:

$$\mathbb{T} = \{D | ID_D mod 4 \equiv 0\} \tag{8.1}$$

Sie enthält demnach alle Relationen deren ID ohne Rest durch vier teilbar ist. Hierdurch ergeben sich 294 Relationen, was 25,8 % der Relationen entspricht. Es handelt sich um eine repräsentative Teilmenge aller Relationen der Basismenge. Dies ist dadurch begründet, dass es sich bei dem für die Berechnung des Modulos relevanten Teil der ID um eine Zufallszahl handelt. Um die Ergebnisse vergleichen zu können, werden zunächst auch für Person A nur die Begutachtungsergebnisse der Teilmenge $\mathbb{T}$ betrachtet. Als Maß zur Beurteilung der Übereinstimmung der Personen A und B wird die Kappa-Statistik nach Cohen (Cohens Kappa) [Coh60] verwendet. Es handelt sich um ein Maß für die sogenannte Interrater-Reliabilität. Anhand der Kappa-Statistik kann die Verlässlichkeit der Beurteilungen der zwei "Rater" A und B eingeschätzt werden. Sie wurde ursprünglich für die Psychologie und Sozialforschung entwickelt, findet jedoch auch in anderen Anwendungsbereichen, wie zum Beispiel der Computer-Lingustik Anwendung [Car96]. Der in dieser Arbeit gegebene Fall manueller Kategorisierung durch zwei Personen ist ein typischer Anwendungsfall für Cohens Kappa. Es ist insbesondere bei Nominalskalen sinnvoll. Zwar können die in Abschnitt 8.1.3 definierten Kategorien zum Teil auch auf eine Ordinalskala, also geordnete Skala, abgebildet werden, eine durchgängige Sortierung der Kategorien ist jedoch in dem gegebenen Fall nicht möglich. Cohens Kappa (κ) ist wie folgt definiert:

$$\kappa = \frac{p_0 - p_c}{1 - p_c} \tag{8.2}$$

Tabelle 8.5.: Bewertungsergebnisse für die Teilmenge $\mathbb{T}$

	Anzahl	Gültig	Gültig - Vorlage	Gültig - Relations-ebene	Gültig - Dokument-ebene	Ungültig
Gesamt	294	84,4 % (248) 82,3 % (242)	10,9 % (32) 12,6 % (37)	3,4 % (10) 3,7 % (11)	1,0 % (3) 0,7 % (2)	0,3 % (1) 0,7 % (2)
Varianten-relationen	156	84,6 % (132) 83,3 % (130)	15,4 % (24) 16,7 % (26)	- -	- -	0,0 % (0) 0,0 %(0)
Element-relationen	127	85,8 % (109) 81,9 % (104)	6,3 % (8) 8,7 % (11)	4,7 % (6) 6,3 % (8)	2,4 % (3) 1,6 % (2)	0,8 % (1) 1,6 % (2)
Aggregations-relationen	11	63,6 % (7) 72,7 % (8)	- -	36,4 % (4) 27,3 % (3)	- -	0,0 % (0) 0,0 %(0)
PowerPoint-Relationen	184	79,9 % (147) 77,2 % (142)	12,5 % (23) 14,7 % (27)	5,4 % (10) 6,0 % (11)	1,6 % (3) 1,1 % (2)	0,5 % (1) 1,1 % (2)
Word-Relationen	110	91,8 % (101) 90,9 % (100)	8,2 % (9) 9,1 % (10)	- -	0,0 % (0) 0,0 % (0)	0,0 % (0) 0,0 % (0)

Hierbei ist p_0 der prozentuale Grad der Übereinstimmung der beiden Beurteilenden, also die Gesamtzahl der übereinstimmenden Beurteilungen dividiert durch die Gesamtzahl der Beurteilungsobjekte. Bei p_c handelt es sich um die erwartete zufällige Übereinstimmung. Sie ist nach Brennan und Prediger definiert als $1/z$, wobei z die Anzahl der Kategorien darstellt [BP81]. In dem hier vorliegenden Fall von fünf möglichen Gültigkeitskategorien gilt also $p_c = 0,20$.

Landis und Koch schlagen in [LK77] eine Einschätzung der Übereinstimmung anhand des Wertes von κ mit Hilfe der in Tabelle 8.4 gegebenen Skala vor. Anhand dieser Skala werden die Ergebnisse im Folgenden bewertet.

8.3.2 Ergebnisse für Basisrelationen

Ergebnisse für die Teilmenge $\mathbb{T}$ der Basisrelationen

Tabelle 8.5 zeigt die Ergebnisse der Begutachtung der beiden Personen für die Teilmenge $\mathbb{T}$. Die Ergebnisse sind jeweils aufgeteilt nach Relations- und Zieldokumenttyp dargestellt. Jede Ergebniszeile besteht aus einer prozentualen und absoluten Darstellung des Ergebnisses für jede der gegebenen Gültigkeitskategorien. Die obere Ergebniszeile in einer Zelle zeigt jeweils das Begutachtungsergebnis von Person A und die untere das von Person B. Aus der Tabelle ist ersichtlich, dass sich die Abweichungen zwischen Person A und Person B überwiegend im Bereich weniger Prozent bewegen. Eine Ausnahme bilden Aggregationsrelationen. Die Abweichungen hier sind jedoch durch die geringe Menge an Beurteilungsobjekten zu erklären. Die Kappa-Statistik bestätigt die Vermutung, dass die Verlässlichkeit der Bewertungen hoch ist: 269 von 294 Beurteilungen stimmen überein. Demnach ergibt sich: $\kappa = 0,894$. Nach Landis und Koch bedeutet dies eine "(fast) vollkommene Übereinstimmung" der Beurteilenden. Es kann also davon ausgegangen werden, dass die Ergebnisse eine hohe Verlässlichkeit besitzen.

Tabelle 8.6.: Kontingenztafel für die Begutachtung

	Gültig	**Vorlage**	**Gültig - Relations-ebene**	**Gültig - Dokument-ebene**	**Ungültig**	**Summe**
Gültig	234	10	4	0	0	248
Vorlage	5	27	0	0	0	32
Gültig - Relationsebene	2	0	6	1	1	10
Gültig - Dokumentebene	1	0	1	1	0	3
Ungültig	0	0	0	0	1	1
Summe	242	37	11	2	2	294

Nichtsdestotrotz ist es interessant, zu analysieren, zwischen welchen Gültigkeitskategorien es die größte Fluktuation gab. Dies kann anhand einer Kontingenztafel ermittelt werden (Tabelle 8.6). Hierbei beziehen sich die Zeilen auf die Beurteilungsergebnisse von Person A und die Spalten auf die von Person B. Jeder Eintrag gibt die Übereinstimmung der beiden Beurteilenden für jede Kategorie an. Die erste Zeile gibt demnach an, wie Person B die von Person A als "Gültig" eingestuften Relationen bewertet hat.

Mit 60 % den größten Anteil an den unterschiedlich beurteilten Relationen hat die Unterscheidung zwischen den Kategorien "Gültig" und "Vorlage". Die Unterschiede traten in beiden Richtungen auf: Sowohl wurden von Person A als gültig bewertete Relationen von Person B als Vorlage eingestuft als auch umgekehrt. Die hohe Fluktuation bei diesen Kategorien ist darin begründet, dass der Beurteilungsspielraum bei der Unterscheidung zwischen Vorlage und gültiger Relation höher ist als bei den anderen Gültigkeitskategorien. Häufig kann es, insbesondere bei Variantenrelationen, zu Grenzfällen kommen, zum Beispiel wenn die Dokumente zwar zu einem Variantenstrang gehören, aber kaum inhaltliche Übereinstimmungen aufweisen. Wenn ein Dokument kopiert wird, um es als Vorlage zu verwenden, werden häufig große Teile des Inhalts der Vorlage gelöscht. Es kommt jedoch vor, dass inhaltiche Teile der Vorlage erhalten bleiben, so dass der Übergang zwischen inhaltlicher Verwandtschaft und reiner Vorlage verwischt. Des Weiteren können unterschiedliche beurteilende Personen eine unterschieldiche Auffassung von inhaltlicher Relevanz haben. Das Logo eines Forschungsprojektes mag für eine Person als inhaltlich relevant gelten, während es die andere Person als rein strukturelles Element betrachtet.

Auch zwischen den Kategorien "Gültig" und "Gültig auf Relationsebene" gibt es Fluktuation. In einigen Fällen wurden sehr kleine Artefakte, zum Beispiel Icons oder kurze Textsequenzen, wiederverwendet. Wenn eine beurteilende Person ein solches Artefakt bei der Bewertung einer Relation übersieht, kommt es zu dieser Art von Verwechslung.

Ergebnisse für die Gesamtmenge der Basisrelationen

Da von einer hohen Verlässlichkeit der Beurteilung der Relationen in der Teilmenge ausgegangen werden kann, kann auch für die gesamte Basismenge eine hohe Verlässlichkeit angenommen werden, auch wenn diese lediglich von Person A gänzlich beurteilt wurde. In Tabelle 8.7 sind die Ergebnisse der Begutachtung der gesamten Basismenge dargestellt. Im Folgenden werden diese erläutert.

Tabelle 8.7.: Evaluationsergebnisse für die Gesamtmenge der Basisrelationen

	Anzahl	Gültig	Gültig - Vorlage	Gültig - Relations-ebene	Gültig - Dokument-ebene	Ungültig
Gesamt	1141	85,5 % (976)	9,6 % (109)	2,5 % (28)	1,4 % (16)	1,1 % (12)
Varianten-relationen	665	87,8 % (584)	12,0 % (80)	-	-	0,2 % (1)
Element-relationen	447	83,9 % (375)	6,5 % (29)	3,8 % (17)	3,6 % (16)	2,2 % (10)
Aggregations-relationen	29	58,6 % (17)	-	37,9 % (11)	-	3,4 % (1)
PowerPoint-Relationen	665	81,2 % (540)	10,8 % (72)	4,2 % (28)	2,1 % (14)	1,7 % (11)
Word-Relationen	475	91,6 % (437)	7,8 % (37)	-	0,4 % (2)	0,2 % (1)

Insgesamt wurde die Validität der Relationen hoch bewertet. Mehr als 85 % der Relationen wurde als gültig ohne Einschränkungen bewertet, während lediglich 1,1 % als vollkommen ungültig bewertet wurden. 10 % der Relationen wurden als Vorlage eingestuft. Diese Relationen sind zwar technisch gesehen gültig, das heißt, das LIS.KOM-System hat die Wiederverwendung korrekt erfasst, die durch eine solche Relation verbundenen Dokumente stehen jedoch in keinem inhaltlichen Zusammenhang. Somit können 95 % der Relationen als aus technischer Sicht gültig angesehen werden. Die inhaltliche Verwandtschaft von durch eine Wiederverwendungsbeziehung verbundenen Dokumenten ist jedoch für viele Anwendungsfälle zur Nutzung der Informationen das ausschlaggebende Kriterium. Die Evaluation des Fingerprinting-Algorithmus MiLe auf den annotierten Variantenrelationen, die sich aus der Berwertung der Relationen ergaben, hat gezeigt, dass als Vorlage eingestufte Relationen mit sehr hoher Verlässlichkeit automatisiert von gültigen Relationen getrennt werden können. Relationen, die Artefaktfehler aufweisen oder zumindest auf Dokumentebene gültig sind, können trotzdem in vielen Szenarien genutzt werden. Häufig wird für eine Nutzung nur die Beziehung auf Dokumentebene benötigt. Somit sind 89 % der erfassten Relationen für einen Großteil der Anwendungsfälle nutzbar.

Variantenrelationen weisen eine sehr hohe Validität auf (88 %). Ungültige Relationen sind in diesem Fall sehr selten. Dies hängt damit zusammen, dass die Erfassung von Variantenrelationen technisch ohne Schwierigkeiten umsetzbar ist. Varianten sind demnach fast grundsätzlich entweder vollständig gültig, oder eine Verwendung des Quelldokuments als Vorlage ist erkennbar. Die Nutzung als Vorlage erfolgt bei Variantenrelationen öfter als bei anderen Relationstypen. Dokumente werden häufig kopiert, um sie als Vorlage für andere Dokumente zu verwenden, auch wenn diese sich inhaltlich stark unterscheiden.

Da die Menge der erfassten Aggregationsrelationen im Vergleich sehr klein ist, Aggregationsrelationen machen nur 3 % aller erfassten Relationen aus, wirken sich Fehler bei der Erfassung hier stärker aus. Bis auf eine ungültige Relation sind jedoch auch die Aggregationsrelationen in den meisten Anwendunsgfällen nutzbar.

Die Gültigkeit der Word-Relationen im Vergleich zu PowerPoint-Relationen zeigt deutlich den Nutzen der Validierung, die bei Word-Relationen durchgeführt wurde. Über 99 % der Word-Relationen wurden technisch gesehen korrekt erfasst. Da bei Variantenrelationen die Validierung keine Anwendung fand, wurde hier nicht zwischen inhaltlich gültigen Relationen und Vorlagen unterschieden. Mit den in Kapitel 6 beschriebenen Techniken ist dies jedoch möglich und die Ergebnisse zeigen, dass eine automatisierte Unterscheidung gute Ergebnisse liefert. Das Verhältnis der als Vorlage eingestuften Relationen zur Gesamtzahl der Word-Relationen ist deutlich geringer als bei PowerPoint.

Es ist wahrscheinlich, dass durch die bei Word-Relationen durchgeführte Validierung, den gegebenen Kriterien nach, gültige Relationen entfernt wurden. So gibt es bei PowerPoint-Relationen einige Fälle, bei denen eine Validierung mittels der in Kapitel 5 und 6 beschriebenen Verfahren zur Entfernung der jeweiligen Relation geführt hätte, beispielsweise weil ein Artefakt oder Dokument kopiert und die Kopie dann übersetzt wurde. Eine auf diese Art und Weise zustande gekommene Relation ist bei der Begutachtung als gültig einzustufen, ein Validierungsalgorithmus hätte sie jedoch mit hoher Wahrscheinlichkeit aufgrund fehlender textlicher Überschneidungen als ungültig markiert.

Die Validität der PowerPoint-Relationen ist zwar insgesamt geringer als die der Word-Relationen, jedoch immer noch ausreichend hoch. Mehr als 80 % der Relationen sind gültig, während lediglich 1,7 % als vollkommen ungültig eingestuft wurden. Dies zeigt, dass auch ohne inhaltliche Validierung, selbst in einem offenen Anwendungsszenario, eine hohe Validität erreicht werden kann.

8.3.3 Ergebnisse für abgeleitete Relationen

Da abgeleitete Relationen nicht eine so hohe Relevanz für die Nutzung besitzen wie Basisrelationen, wurden diese lediglich von Person A beurteilt. Hier wurden aufgrund der großen Anzahl nur Relationen beurteilt, für deren ID gilt: $ID \mod 8 \equiv 0$. Es ergab sich eine zu begutachtende Menge von 373 Relationen. Damit kann eine tendenzielle Aussage über die Gültigkeit der abgeleiteten Relationen getroffen werden. Die Ergebnisse sind in Tabelle 8.8 wiederum anhand der Relations- sowie Zieldokumenttypen aufgeteilt.

Die abgeleiteten Relationen bieten ein ähnliches Bild wie die Basisrelationen. Hier ist zu erwarten, dass die Zahl der gültigen PowerPoint-Relationen im Vergleich zur Basismenge geringer ist, da viele Aktionen im Lebenszyklus eines Dokuments nicht nachverfolgt werden konnten. Diese Vermutung wird durch die Ergebnisse der Beurteilung auch bestätigt (siehe Abbildung 8.8), obgleich die Unterschiede zur Basismenge vergleichsweise klein ausfallen. Ohne die im Nachhinein vorgenommene naive Validierung der PowerPoint-Relationen, wäre die Anzahl der ungültigen oder nur begrenzt gültigen PowerPoint-Relationen wesentlich höher gewesen. Die Zahlen zeigen jedoch auch, dass eine einfache strukturelle Validierung für PowerPoint-Relationen ausreichend ist, um eine hohe Gültigkeit zu gewährleisten.

Die Validierung der Word-Relationen hat nicht nur eine starke Verringerung der vererbten Relationen zur Folge, sondern stellt auch deren Validität sicher: Alle abgeleiteten Word-Relationen wurden bei der Bewertung als gültig eingestuft. Erwartungsgemäß finden sich die in der Basismenge vorhandenen Fehler bei Aggregationsrelationen in der Menge der abgeleiteten Relationen wieder.

Tabelle 8.8.: Evaluationsergebnisse für abgeleitete Relationen

	Anzahl	Gültig	Gültig - Vorlage	Gültig - Relations-ebene	Gültig - Dokument-ebene	Ungültig
Gesamt	373	79,9 % (298)	5,4 % (20)	11,0 % (41)	1,9 % (7)	1,9 % (7)
Element-relationen	354	82,5 % (292)	5,6 % (20)	8,2 % (29)	1,7 % (6)	2,0 % (7)
Aggregations-relationen	19	31,6 % (6)	-	63,2 % (12)	-	0,0 % (0)
PowerPoint-Relationen	359	79,1 % (284)	5,6 % (20)	11,4 % (41)	1,9 % (7)	1,9 % (7)
Word-Relationen	13	100 % (13)	0,0 % (0)	-	0,0 % (0)	0,0 % (0)

8.4 Zusammenfassung und Fazit

Die Evaluation des LIS.KOM-Frameworks hat gezeigt, dass eine Erfassung von Beziehungsinformationen aus dem Lebenszyklus von Wissensdokumenten mit hoher Verlässlichkeit möglich ist. Beide vorgestellten Verfahren für die Gewährleistung der Gültigkeit der erfassten Relationen weisen gute Ergebnisse auf. In beiden Fällen sind die erfassten Relationen in über 87 % der Fälle nutzbar, das heißt, die in Relation stehenden Dokumente weisen einen tatsächlichen inhaltlichen Zusammenhang auf. Dies ist bei allen Gültigkeitskategorien gegeben, außer bei ungültigen Relationen und Vorlagen. Die zur Laufzeit durchgeführte Validierung der Word-Relationen hat eine sehr hohe Gültigkeit der erfassten Relationen zur Folge. In diesem Fall kann jedoch nicht ausgeschlossen werden, dass auch gültige Relationen durch die Validierung entfernt werden, wenn die inhaltliche Beziehung etwa nicht mit einer Überschneidung auf Wortebene einhergeht, wie es beispielsweise bei Übersetzungen der Fall ist.

Eine vollständige Nachverfolgung aller PowerPoint-Relationen konnte aufgrund der Offenheit des Evaluationsszenarios nicht gewährleistet werden. Der erwartete negative Effekt konnte jedoch durch eine naive Validierung der Relationen im Nachhinein ausreichend abgemildert werden. So ergab sich auch bei PowerPoint-Relationen einen hohe Gültigkeitsquote. Mit den in Kapitel 6 vorgestellten Verfahren ist es weiterhin möglich, im Fall von Variantenrelationen automatisiert gültige Relationen von denen als Vorlage oder ungültig eingestuften Wiederverwendungsrelationen zu unterscheiden. So kann die Quote der nutzbaren Relationen auf über 90 % erhöht werden. Nur in weniger als 2 % der Fälle wurden komplett ungültige Relationen erfasst.

Auch die quantitative Auswertung brachte relevante Ergebnisse. Zunächst bestätigte sich durch die Evaluation die eingangs der Arbeit getätigte Annahme, dass Wiederverwendung ein häufig auftretender Prozess im Lebenszyklus von Wissensdokumenten ist. Die Evaluation hat gezeigt, dass Wiederverwendungsrelationen in großem Umfang auftreten und erfasst werden können. PowerPoint-Dokumente weisen eine deutlich stärkere Wiederverwendung auf Elementebene auf und besitzen eine größere Anzahl von Relationen pro Dokument, sind also stärker untereinander vernetzt als Word-Dokumente. Durch

die Validierung der Word-Relationen ergab sich eine deutlich geringere Zahl an durch Variantenbildung abgeleiteten Relationen als bei PowerPoint.

Die durchgeführte Evaluation konzentriert sich allein auf die Erfassung der Relationen. Es wurde gezeigt, dass die Erfassung durchführbar ist und überwiegend gültige Ergebnisse liefert. Dadurch lässt sich jedoch keine Aussage über die Nutzbarkeit oder Nützlichkeit der erfassten Relationen treffen. Um dies beurteilen zu können, müssen die erfassten Relationen in einem geeigneten Nutzungsszenario eingesetzt und entsprechend evaluiert werden. Das ist jedoch kein Ziel dieser Arbeit.

8.5 Bewertung des Gesamtsystems

Die Evaluation des LIS.KOM-Frameworks über mehrere Monate hinweg hat gezeigt, dass das umgesetzte System funktions- und lauffähig ist. Nach dem Beginn der Evaluation und während der Laufzeit traten keine für die Lauffähigkeit kritischen technische Probleme auf - das LIS.KOM-Framework arbeitete stabil. Die Tatsache, dass bei allen Nutzern auch nach der initial geplanten achtwöchigen Laufzeit der Erfassungs-Plug-ins weiterhin Relationen erfasst wurden, zeigt, dass die Erfassungs-Plug-ins über einen längeren Zeitraum stabil funktionieren und keine signifikanten Behinderungen der Arbeit der Teilnehmer verursachen. Es wurde gezeigt, dass die Menge der Relationen ein lineares Wachstum aufweist, so dass die Skalierbarkeit des LIS.KOM-Frameworks bezüglich der Verwaltung der erfassten Beziehungsinformationen gegeben ist. Zudem wurden verschiedene Strategien zur Verringerung der Menge der vorgehaltenen Relationen sowie zur Reduzierung redundanter Informationen erarbeitet (Kapitel 4.2.4). Diese können umgesetzt werden, sollte die Skalierbarkeit zur Herausforderung werden. Des Weiteren wurden zwei verschiedene Nutzungsszenarien entworfen und prototypisch umgesetzt. Zum einen die Unterstützung der Nutzer von Wissensdokumenten während des Autorenprozesses und zum anderen die Unterstützung des Retrievals von Wissensdokumenten durch eine Erweiterung des Dateisystem-Explorers. Die Machbarkeit und Funktionsfähigkeit des Systems wurde somit in allen Bereichen von der Erfassung über die Verwaltung bis hin zur Nutzung im Sinne eines Proof of Concept gezeigt.

Zum Abschluß erfolgt eine Beurteilung des umgesetzten und evaluierten LIS.KOM-Frameworks anhand der in Kapitel 2.5 gestellten Anforderungen an ein System zur Erfassung von Lebenszyklusinformationen:

1. *Die Informationen müssen dort erfasst werden, wo sie entstehen:* Es wurden Plug-ins für PowerPoint und Word - zwei der häufigsten Formate für Wissensdokumente - umgesetzt, welche die Informationen dort erfassen, wo sie entstehen. Es wurde zudem gezeigt, dass das Konzept auch einfach auf andere Applikationen übertragbar ist. Die Anforderungen an eine solche Applikation für die verlässliche Erfassung von Informationen in einer solchen Applikation wurden ermittelt. Es hat sich zudem gezeigt, dass diese von den meisten Applikationen erfüllt werden können.

2. *Wiederverwendung muss berücksichtigt werden:* Das LIS.KOM-Framework in der Form, in der es umgesetzt wurde, konzentriert sich überwiegend auf die Erfassung von Wiederverwendungsbeziehungen.

3. *Die Informationen müssen systemübergreifend bereitgestellt werden:* Die Informationen werden mit Hilfe des LIS.KOM-Servers nach der Erfassung anderen Systemen zur Verfügung gestellt und kön-

nen über eine generische Webservice-Schnittstelle abgerufen werden. Dadurch ist gewährleistet, dass die Informationen nach der Erfassung nicht in dem entsprechenden System, beziehungsweise der Applikation verbleiben, sondern auch von anderen Nutzern und Systemen zugegriffen werden können.

4. *Informationen müssen genutzt werden können:* Es wurde ein strukturiertes Format zur Verwaltung von Lebenszyklusinformationen entworfen und genutzt, das erweiterbar und kompatibel zu bestehenden Formaten im Bereich der Erfassung von Lebenszyklusinformationen ist.

5. *Die Erfassung sollte den gesamten Lebenszyklus eines Dokumentes abdecken:* Mit dem LIS.KOM-Framework ist es möglich, den gesamten Lebenszyklus eines Dokumentes abzudecken. Dies wurde mit Hilfe der drei Erfassungskomponenten für PowerPoint, Word sowie das Windows Dateisystem umgesetzt. Es müssen lediglich die entsprechenden Plug-ins bzw. Applikationen auf den am Lebenszyklus eines Wissensdokuments beteiligten Systemen installiert sein. Die Evaluation hat zudem gezeigt, dass in einem offenen Szenario, in dem dies nicht gewährleistet werden kann, die Erfassung dennoch sinnvolle Ergebnisse liefert.

6. *Die erfassten Informationen sollten verlässlich sein:* Die Evaluation des LIS.KOM-Frameworks hat gezeigt, dass einen Erfassung mit hoher Verlässlichkeit möglich ist - die erfassten Relationen also eine hohe Gültigkeit aufweisen. Mit Hilfe der entwickelten Validierungsalgorithmen ist es möglich, die Verlässlichkeit der erfassten Informationen zu gewährleisten und sie sogar noch zu erhöhen.

Das LIS.KOM-Framework erfüllt demnach die gestellten Anforderungen in vollem Umfang.

9 Zusammenfassung und Ausblick

Das abschließende Kapitel gibt eine Zusammenfassung der Hauptbeiträge der Arbeit. Es folgt ein Ausblick und die Identifikation von Ansatzpunkten für weitere Arbeiten, die sich durch die vorliegende Arbeit ergeben haben.

9.1 Zusammenfassung

Die vorliegende Arbeit hatte zum Ziel, durch die automatisierte Erfassung von Lebenszyklusinformationen im Allgemeinen und Beziehungsinformationen im Besonderen, eine Grundlage und Voraussetzung für die Nutzung dieser Informationen in diversen Anwendungsszenarien zu schaffen. Dieses Ziel konnte mit Hilfe folgender Beiträge erreicht werden:

Lebenszyklusinformationen für Wissensdokumente wurden definiert, kategorisiert und identifiziert. Dies geschah anhand eines zu diesem Zweck entwickelten *Lebenszyklusmodells*. Anhand bestehender Definitionen und Klassifikationen wurden Lebenszyklusinformationen und Beziehungsinformationen definiert und verschiedene Typen von Wiederverwendungsbeziehungen hergeleitet. Die zugrundeliegenden Prozesse und die Bedingungen, unter welchen diese auftreten, wurden ermittelt und beschrieben.

Darauf basierend wurde das *LIS.KOM-Schema* zur Speicherung von Beziehungsinformationen entwickelt und erfolgreich im LIS.KOM-Framework eingesetzt. Es ist kompatibel zum bestehenden CAM-Schema, das der Verwaltung von Verwendungsinformationen dient. Dadurch können CAM-Metadaten mit erfassten Beziehungsinformationen erweitert werden.

Das *LIS.KOM-Framework* zur Erfassung, Verwaltung und Nutzung von strukturierten Lebenszyklusinformationen für Wissensdokumente wurde konzipiert, implementiert und evaluiert. Besonderes Gewicht lag dabei auf der Erfassung von Beziehungsinformationen. Hierfür wurde ein auf Plug-ins basierendes Konzept entwickelt und für zwei der meist genutzten Applikationen zur Verarbeitung von lokalen Wissensdokumenten umgesetzt. Die Minimalanforderungen für eine Übertragbarkeit des Konzepts auf andere Applikationen wurden analysiert und erläutert. Zudem wurde eine Erfassungskomponente für das Windows-Dateisystem umgesetzt. Somit ist für die unterstützten Dokumente der komplette Lebenszyklus abgedeckt. Es wurden zwei unterschiedliche *Ansätze zur Gewährleistung der Gültigkeit* der erfassten Informationen konzipiert und anhand der beiden Applikationen Word und PowerPoint implementiert und verglichen. Zudem wurde eine nutzerbasierte Evaluation der Gültigkeit der erfassten Relationen durchgeführt. Sie hat gezeigt, dass in beiden Fällen die Erfassung von Informationen mit genügend hoher Gültigkeit durchführbar ist. Es wurde ein System zur Verwaltung und Bereitstellung der erfassten Informationen entwickelt, welches sich in der Evaluation als robust und verlässlich herausgestellt hat. Die durchgeführte Evaluation gewährt zudem Einblicke in das Wiederverwendungsverhalten von Nutzern von PowerPoint- und Word-Dokumenten und erlaubt Vergleiche zwischen beiden Typen zu ziehen.

Für die *Validierung* erfasster Beziehungsinformationen wurden zwei verschiedene Algorithmen entwickelt: *ShingleCloud*, ein String-Matching-Ansatz, und MiLe, ein Fingerprinting-Algorithmus. Shingle-

Cloud zeigt in den meisten getesteten Fällen eine bessere Qualität oder ein deutlich besseres Laufzeitverhalten bei vergleichbarer Qualität als die Referenzalgorithmen. Er eignet sich zudem besser als die Vergleichsalgorithmen dazu, übereinstimmende Textstellen aufzufinden und zu markieren. Im LIS.KOM-Framework wurde er erfolgreich zur Validierung von Elementrelationen sowie zur Unterstützung der Evaluation eingesetzt. Der Ansatz eignet sich prinzipiell für alle Szenarien, in denen eine Nadel-Heuhaufen-Suche durchgeführt werden soll, dazu zählen Text-Retrieval oder Plagiatssuche.

Der *Fingerprinting-Ansatz MiLe* schneidet ebenso auf fast allen Korpora besser ab als der Referenzalgorithmus K-Gram, bei deutlich geringerem Speicherverbrauch. Für laufzeitkritische Szenarien wurde ein Kompressionsverfahren entwickelt, das wiederum bessere Ergebnisse aufweist als vergleichbare Kompressionsverfahren. Der auf textbasierte Dokumente ausgerichtete Original-Algorithmus wurde zudem erweitert und auf objektbasierte Dokumente anwendbar gemacht. Sowohl die textbasierte als auch die objektbasierte Form des Algorithmus wurden zusätzlich auf den im LIS.KOM-Framework gesammelten Daten evaluiert, und es hat sich gezeigt, dass beide sehr gut für die automatisierte Beurteilung der Gültigkeit von Variantenrelationen geeignet sind. Die weiteren Einsatzmöglichkeiten des Algorithmus sind vielfältig: Da das Abdeckungstupel zweier Dokumente einzig auf Basis des MiLe-Fingerprints und ohne erneuten Zugriff auf den Inhalt der Dokumente berechnet werden kann, eignet er sich besonders gut für datenschutz- oder informationssicherheitskritische Anwendungen. So kann eine Ähnlichkeits- oder Plagiatssuche erfolgen ohne Preisgabe des Inhalts eines Dokuments. Zudem können Übereinstimmungen auch allein auf Basis des Fingerprints im Quell- oder Zieldokument lokalisiert werden.

9.2 Ausblick auf zukünftige Arbeiten

Es gibt verschiedene Ansatzpunkte für weitere Arbeiten am LIS.KOM-Framework: Lebenszyklusinformationen müssen dort erfasst werden, wo sie entstehen. Deshalb verfolgt das LIS.KOM-Framework ein auf Plug-ins basierendes Konzept. Eine Erfassung von Informationen in einer Applikation bzw. einem System setzt voraus, dass ein entsprechendes Plug-in vorhanden ist. In der Arbeit wurden drei Erfassungskomponenten umgesetzt. Ein weiterer Schritt kann die Umsetzung weiterer Plug-ins für unterschiedliche Systeme und Applikationen wie zum Beispiel Dokumentenmanagementsysteme, Repositories, Contentmanagement-Systeme oder auch andere Office-Applikationen wie Tabellenkalkulations-Programme oder Modellierungswerkzeuge sein. So kann der Lebenszyklus von Wissensdokumenten noch besser und für noch mehr Typen von Dokumenten abgedeckt werden.

Ein weiterer Ansatzpunkt für Forschungsarbeiten stellt die Verarbeitung von Lebenszyklusinformationen dar. Die beschriebenen Methoden zur Gewichtung, Anreicherung und Konsolidierung von Lebenszyklusinformationen wurden zwar zum Teil umgesetzt, aber nicht evaluiert oder verglichen. Auf diesem Gebiet gibt es jedoch einige interessante Fragestellungen: Wie sind Parameter bei der Gewichtung von Relationen zu wählen, wie können Verwendungsinformationen mit Beziehungsinformationen angereichert werden und umgekehrt, wie sind die Schwellwerte für eine Konsolidierung von Beziehungsinformationen zu wählen und wie stark wirkt sich der gegebene Informationsverlust auf die Nutzbarkeit aus. Auch die Differenzierung von Relationen als weiterer Verarbeitungsschritt, insbesondere im Fall von Variantenrelationen, stellt eine sinnvolle Erweiterung dar. Im momentanen Zustand unterscheidet das LIS.KOM-Framework keine unterschiedlichen Subtypen bei Variantenrelationen. Dies kann aber durch-

aus sinnvoll sein, da der Typ Variantenrelation ein weites Spektrum von Beziehungen zwischen Dokumenten abdeckt (siehe Kapitel 2).

Die oben genannten Fragen können nur im Zusammenhang mit entsprechenden Nutzungsszenarien beantwortet werden. Dies ist gleichzeitig der wichtigste Ansatzpunkt für weitere Arbeiten. Die vorliegende Arbeit hat gezeigt, dass und wie strukturierte Lebenszyklusinformationen verlässlich erfasst werden können. Die erfassten Informationen müssen nun zur Unterstützung der Nutzer der Wissensdokumente eingesetzt werden. Die vorliegende Arbeit konnte hierfür zwar Vorschläge liefern und prototypisch implementieren, diese wurden aber nicht produktiv genutzt oder evaluiert. Um die Nützlichkeit von Lebenszyklusinformationen beurteilen und evaluieren zu können, muss ein entsprechendes Szenario entwickelt werden, anhand dessen die erfassten Informationen und ihr Nutzen bewertet werden können.

Eine weitere wichtige Herausforderung, die sich grundsätzlich stellt, wenn personalisierte Daten erfasst werden, ist die Frage nach Datenschutz und der Sicherheit der erfassten Daten. Diese Bereiche wurden aus der vorliegenden Arbeit bewusst ausgeklammert, da ihre Behandlung eine Vielzahl rechtlicher und die Privatsphäre betreffender Fragen aufwirft, die den inhaltlichen Rahmen dieser Dissertation gesprengt hätten.

Literaturverzeichnis

[20n09] The 20 newsgroups data set. Online: http://people.csail.mit.edu/jrennie/20Newsgroups/, Zuletzt abgerufen am 16.11.2009.

[AD06] David W. Archer and Lois M. L. Delcambre. Capturing and reusing human attention in corporate decision making. In *CAMA '06: Proceedings of the 1st international workshop on Contextualized attention metadata: collecting, managing and exploiting of rich usage information*, pages 17–20, New York, NY, USA, 2006. ACM.

[Ada84] John G. Adair. The hawthorne effect: a reconsideration of the methodological artifact. *Journal of applied psychology*, 69:334–345, 1984.

[ADL04] Advanced Distributed Learning Initiative ADL. Sharable content object reference model (scorm), 4th edition. Online: http://www.adlnet.gov/Technologies/scorm/SCORMSDocuments/2004n/Overview.aspx, 2004. Zuletzt abgerufen am 17.11.2009.

[AKKW04] Apostolos Antonacopoulos, Dimosthenis Karatzas, Henryk Krawczyk, and Bogdan Wiszniewski. The lifecycle of a digital historical document: structure and content. In *DocEng '04: Proceedings of the 2004 ACM symposium on Document engineering*, pages 147–154, New York, NY, USA, 2004. ACM.

[Alb08] Badawia M. Albassuny. Automatic metadata generation applications; a survey study. *Int. J. Metadata Semant. Ontologies*, 3(4):260–282, 2008.

[ANR74] N. Ahmed, T. Natarajan, and K.R. Rao. Discrete cosine transform. *IEEE Transactions on Computers*, C-23:90– 93, January 1974.

[ARI09] ARIADNE. Ariadne - foundation for the european knowledge pool. Online: http://ariadne-eu.org, Zuletzt abgerufen am 16.11.2009.

[Bae98] Walter R. Baets. *Organization Learning and Knowledge Technologies in a Dynamic Environment*. Kluwer Academic Publishers, Norwell, MA, USA, 1998.

[BCC+06] Ingo Brunkhorst, Paul Alexandru Chirita, Stefania Costache, Julien Gaugaz, Ekaterini Ioannou, Tereza Iofciu, Enrico Minack, Wolfgang Nejdl, and Raluca Paiu. The beagle++ toolbox: Towards an extendable desktop search architecture. In Stefan Decker, Jack Park, Leo Sauermann, Soeren Auer, and Siegfried Handschuh, editors, *Proceedings of the Semantic Desktop and Social Semantic Collaboration Workshop (SemDesk 2006) at the 5th International Semantic Web Conference ISWC 2006*, volume 202 of *CEUR-WS.org*, nov 2006.

[BCN06] Jürgen Belizki, Stefania Costache, and Wolfgang Nejdl. Application independent metadata generation. In *CAMA '06: Proceedings of the 1st international workshop on Contextualized*

attention metadata: collecting, managing and exploiting of rich usage information, pages 33–36, New York, NY, USA, 2006. ACM.

[BCR09] Alberto Barrón-Cede no and Paolo Rosso. On automatic plagiarism detection based on n-grams comparison. In *ECIR '09: Proceedings of the 31th European Conference on IR Research on Advances in Information Retrieval*, pages 696–700, Berlin, Heidelberg, 2009. Springer-Verlag.

[BDGM95] Sergey Brin, James Davis, and Héctor García-Molina. Copy detection mechanisms for digital documents. In *SIGMOD '95: Proceedings of the 1995 ACM SIGMOD international conference on Management of data*, pages 398–409, New York, NY, USA, 1995. ACM. Hash Breaking.

[BHK98] John S. Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. Technical report, Microsoft Research, 1998.

[BHL+07] Sonja Bergsträßer, Tomas Hildebrandt, Lasse Lehmann, Christoph Rensing, and Ralf Steinmetz. Virtual context based services for support of interaction in virtual worlds. In Grenville Armitage, editor, *Netgames 2007, 6th Annual Workshop on Network and Systems Support for Games, Melbourne, Australia*, pages 111–116, Swinburne University of Technology, Australia, Sep 2007. ACM SIGCOMM.

[BHRS09] Sonja Bergsträßer, Tomas Hildebrandt, Christoph Rensing, and Ralf Steinmetz. Virtual context based services for multiplayer online games to facilitate community participation. *Multimedia Tools and Applications*, May 2009.

[BKJH10] Holger Brocks, Alfred Kranstedt, Gerald Jäschke, and Matthias Hemmje. *Smart Information and Knowledge Management: Advances, Challenges, and Critical Issues*, volume 260, chapter Modeling Context for Digital Preservation, pages 197–226. Springer Berlin / Heidelberg, 2010.

[BM90] David C. Blair and M. E. Maron. Full-text information retrieval: further analysis and clarification. *Inf. Process. Manage.*, 26(3):437–447, 1990.

[BM06] Christopher Brooks and Gord McCalla. Towards flexible learning object metadata. In *Proceedings of Int. J. Cont. Engineering Education and Lifelong Learning, Vol16, Nos 1/2*, 2006.

[BP81] Robert L. Brennan and Dale J. Prediger. Coefficient kappa: Some uses, misuses, and alternatives. *Educational and psychological measurement*, 41(3):687–699, 1981.

[Bro97] Andrei Z. Broder. On the resemblance and containment of documents. In *SEQUENCES '97: Proceedings of the Compression and Complexity of Sequences 1997*, page 21, Washington, DC, USA, 1997. IEEE Computer Society.

[Bro00] Andrei Z. Broder. Identifying and filtering near-duplicate documents. In *COM '00: Proceedings of the 11th Annual Symposium on Combinatorial Pattern Matching*, pages 1–10, London, UK, 2000. Springer-Verlag.

[BYRN99] Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, May 1999.

[CAM07] CAMs. Contextualized attention metadata conceptual base scheme. Online: http://ariadne.cs.kuleuven.ac.be/empirical/attention/CAMv1.5.pdf, 2007. Zuletzt abgerufen am 16.11.2009.

[Car96] Jean Carletta. Assessing agreement on classification tasks: the kappa statistic. *Comput. Linguist.*, 22(2):249–254, 1996.

[Car07] Kris Cardinaels. *A Dynamic Learning Object Life Cycle and its Implications for Automatic Metadata Generation*. PhD thesis, KATHOLIEKE UNIVERSITEIT LEUVEN, 2007.

[CCNP06] Paul-Alexandru Chirita, Stefania Costache, Wolfgang Nejdl, and Raluca Paiu. Beagle++: Semantically enhanced searching and ranking on the desktop. In *The Semantic Web: Research and Applications*, volume Volume 4011/2006 of *Lecture Notes in Computer Science*, pages 348–362. Springer Berlin / Heidelberg, 2006.

[CGG+05] Paul-Alexandru Chirita, Rita Gavriloaie, Stefania Ghita, Wolfgang Nejdl, and Raluca Paiu. Activity based metadata for semantic desktop search. In *Proceedings of the 2nd European Semantic Web Conference*, pages 439–454, Heraklion, Crete, Greece, May 2005.

[CGPW02] Paul Clough, Robert Gaizauskas, Scott S.L. Piao, and Yorick Wilks. METER: MEasuring TExt Reuse. In *Proceedings of the 40th Anniversary Meeting for the Association for Computational Linguistics (ACL-02)*, pages 152–159, Philadelphia, July 2002.

[Cha02] Moses S. Charikar. Similarity estimation techniques from rounding algorithms. In *STOC '02: Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 380–388. ACM Press, 2002.

[Chi07] Paul Alexandru Chirita. *Emerging Applications of Link Analysis for Ranking*. PhD thesis, Fakultaet fuer Elektrotechnik und Informatik der Gottfried Wilhelm Leibniz Universitaet Hannover, 2007.

[Clo03] Paul Clough. Old and new challenges in automatic plagiarism detection. Online: http://ir.shef.ac.uk/cloughie/papers/pas_plagiarism.pdf, 2003. (zuletzt abgerufen am 16.11.2009).

[CM09a] James Cummings and Arno Mittelbach. The holinshed project: Comparing and linking two editions of holinshed's chronicles. In Sue Broadhurst, editor, *Proceedings of Digital Resources for the Humanities and Arts*, 2009.

[CM09b] James Cummings and Arno Mittelbach. The tei-comparator: A web-based comparison tool for xml editions. In *Proceedings of Text encoding in the era of mass digitization Conference and Members' Meeting of the TEI Consortium*, 2009.

[CN06] Paul-Alexandru Chirita and Wolfgang Nejdl. Analyzing user behavior to rank desktop items. In Fabio Crestani, Paolo Ferragina, and Mark Sanderson, editors, *SPIRE*, volume 4209 of *Lecture Notes in Computer Science*, pages 86–97. Springer, 2006.

[Coh60] Jacob Cohen. A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, 20(1):37, 1960.

[CVB06] Olivier Catteau, Philippe Vidal, and Julien Broisin. A generic representation allowing for expression of learning object and metadata lifecycle. In *ICALT '06: Proceedings of the Sixth IEEE International Conference on Advanced Learning Technologies*, pages 30–32, Washington, DC, USA, 2006. IEEE Computer Society.

[DCM97] Dublin Core Metadata Initiative DCMI. Request for comment (rfc) 2413: Relation element working draft (1997). Online: http://dublincore.org/documents/relation-element/, 1997. Zuletzt abgerufen am 16.11.2009.

[DCM08] Dublin Core Metadata Initiative DCMI. Iso standard 15836-2003: The dublin core metadata element set. Online: http://dublincore.org/documents/dces/, 2008. Zuletzt abgerufen am 17.11.2009.

[DFC+01] Erik Duval, Eddy Forte, Kris Cardinaels, Bart Verhoeven, Rafael Van Durm, Koen Hendrikx, Maria Wentland Forte, Norbert Ebel, Maciej Macowicz, Ken Warkentyne, and Florence Haenni. The ariadne knowledge pool system. *Commun. ACM*, 44(5):72–78, 2001.

[DH03] Erik Duval and Wayne Hodgins. A lom research agenda. In *Proceedings of the twelfth international conference on World Wide Web*, Budapest, Hungary, 2003.

[DP99] Thomas H. Davenport and Laurence Prusak. *Working Knowledge: How Organizations Manage What They Know*. Harvard Business School Press, 1999.

[DSS93] Randall Davis, Howard E. Shrobe, and Peter Szolovits. What is a knowledge representation? *AI Magazine*, 14(1):17–33, 1993.

[EBJ+09] Felix Engel, Holger Brocks, Gerald Jäschke, Claus-Peter Klas, Alfred Kranstedt, and Matthias Hemmje. Towards supporting context-oriented information retrieval in a scientific-archive based information lifecycle. 2009.

[ECM09] ECMA. Csharp language specification. Online: http://www.ecma-international.org/publications/standards/Ecma-334.htm, Zuletzt abgerufen am 17.11.2009.

[EHS05] Michael Engelhardt, Arne Hildebrand, and Thomas C. Schmidt. Automatisierte augmentierung von lernobjekten in einer semantischen interpretationsschicht der hylos plattform. In Jörg M. Haake, Ulrike Lucke, and Djamshid Tavangarian, editors, *DeLFI 2005 3. Deutsche e-Learning Fachtagung Informatik*, volume 66, pages 105–116, September 2005.

[EKH09] Felix Engel, Claus-Peter Klas, and Matthias Hemmje. Relevance feedback based on context information. In *Proc. of the Information Retrieval 2009 Workshop at LWA 2009, Darmstadt, Germany*, 2009.

[EKR+03] Michael Engelhardt, Anreas Karparti, Torsten Rack, Ivette Schmidt, and Thomas C. Schmidt. Hypermedia Learning Objects System – On the Way to a Semantic Educational Web. In Michael E. Auer and Ursula Auer, editors, *Proceedings of the International Workshop "Interactive Computer aided Learning" ICL 2003. Learning Objects and Reusability of Content*. Kassel University Press, January 2003.

[Faa04] Andreas Faatz. *Ein Verfahren zur Anreicherung fachgebiesspezifischer Ontologien durch Begriffsvorschläge*. PhD thesis, TU Darmstadt, Nov 2004.

[Fel98] Christiane Fellbaum. *WordNet: An Electronic Lexical Database (Language, Speech, and Communication)*. The MIT Press, May 1998.

[Fer03] Reginald Ferber. *Information Retrieval. Suchmodelle und Data-Mining-Verfahren für Textsammlungen und das Web*. Dpunkt Verlag, 1 edition, 2003.

[GASS04] Manuel Goertz, Ralf Ackermann, Johannes Schmitt, and Ralf Steinmetz. Context-aware communication services: A framework for building enhanced ip telephony services. In *International Conference on Computer Communications and Networks (ICCCN) 2004*, pages 272–279, Oct 2004.

[GCC+08] Julien Gaugaz, Stefania Costache, Paul-Alexandru Chirita, Claudiu Firan, and Wolfgang Nejdl. Activity based links as a ranking factor in semantic desktop search. In *LA-WEB '08: Proceedings of the 2008 Latin American Web Conference*, pages 49–57, Washington, DC, USA, 2008. IEEE Computer Society.

[GFW+01] Robert Gaizauskas, Jonathan Foster, Yorick Wilks, John Arundel, Paul Clough, and Scott Piao. The Meter corpus: A corpus for analysing journalistic text reuse. In *Proceedings of Corpus Linguistics 2001 Conference*, pages 214–223. Lancaster University Centre for Computer Corpus Research on Language, 2001.

[GG03] Ana Gil and Francisco García. E-commerce recommenders: powerful tools for e-business. *Crossroads*, 10(2):6–6, 2003.

[GHM+07] Tudor Groza, Siegfried Handschuh, Knud Moeller, Gunnar Grimnes, Leo Sauermann, Enrico Minack, Cedric Mesnage, Mehdi Jazayeri, Gerald Reif, and Rosa Gudjonsdottir. The nepomuk project - on the way to the social semantic desktop. In Tassilo Pellegrini and Sebastian Schaffert, editors, *Proceedings of I-Semantics' 07*, pages pp. 201–211. JUCS, 2007.

[Gil08] Anne J. Gilliland. *Introduction to Metadata*, chapter Setting the Stage, pages 1–19. Getty Research Insitute, Los Angeles, online edition, version 3.0 edition, 2008.

[Gin99] Mark Ginsburg. An agent framework for intranet document management. *Autonomous Agents and Multi-Agent Systems*, 2:271 – 286, September 1999.

[Goe05] Manuel Goertz. *Effiziente Effiziente Echtzeitkommunikationsdienste durch Einbeziehung von Kontexten*. PhD thesis, TU Darmstadt, Fachbereich Elektrotechnik und Informationstechnik, 2005.

[Gru86] Dick Grune. Concurrent versions system, a method for independent cooperation. Technical report, IR 113, Vrije Universiteit, 1986.

[GSK+99] Nathaniel Good, J. Ben Schafer, Joseph A. Konstan, Al Borchers, Badrul Sarwar, Jon Herlocker, and John Riedl. Combining collaborative filtering with personal agents for better recommendations. In *AAAI/IAAI*, 1999.

[GSMP09] Amit Kumar Goel, Ritu Sindhu, Monica Mehrotra, and G N Purohit. Managing unstructured data using agent technology. *Ubiquitous Computing and Communications*, 4 No. 3:801–806, 2009.

[Ham50] Richard W. Hamming. Error detecting and error correcting codes. *Bell System Technical Journal*, 26(2):147–160, 1950.

[HBCH09] Ossama A. Hamid, Behshad Behzadi, Stefan Christoph, and Monika Henzinger. Detecting the origin of text segments efficiently. In *WWW '09: Proceedings of the 18th international conference on World wide web*, pages 61–70, New York, NY, USA, 2009. ACM.

[HBD04] Thomas B. Hodel, Dominik Businger, and Klaus R. Dittrich. Supporting collaborative layouting in word processing. In *CoopIS/DOA/ODBASE (1)*, pages 355–372, 2004.

[Hen06] Monika Henzinger. Finding near-duplicate web pages: A large-scale evaluation of algorithms. In *Proceedings of SIGIR '06*, Seattle, Washington, USA, August 2006. ACM, ACM Press.

[HGM09] Erik Hatcher, Otis Gospodnetic, and Mike McCandless. *Lucene in Action*. Manning Pubn, 0002 edition, 2009.

[HHD05] Thomas Hodel, Roger Hacmac, and Klaus R. Dittrich. Using text editing creation time meta data for document management. In *Proceedings of Conference on Advanced Information Systems Engineering*, pages 105–118, Porto, Portugal, 2005.

[HHRS05] Stefan Hoermann, Tomas Hildebrandt, Christoph Rensing, and Ralf Steinmetz. Resourcecenter - a digital learning object repository with an integrated authoring tool. In *Proceedings of World Conference on Educational Multimedia, Hypermedia*, 2005.

[Hoe05] Stefan Hoermann. *Wiederverwendung von digitalen Objekten in einem auf Aggregation basierenden Autorenprozess*. PhD thesis, Technische Universitaet Darmstadt, 2005.

[Hol09] The holinshed project. Online: http://www.cems.ox.ac.uk/holinshed/about.shtml, Zuletzt abgerufen am 16.11.2009.

[IEE02] IEEE. Ieee standard for learning object metadata 1484.12.1., 2002. http://ltsc.ieee.org/wg12/par1484-12-1.html (zuletzt abgerufen am 16.11.2009).

[Ini09] Text Encoding Initiative. Tei standard. Online: http://www.tei-c.org/Guidelines/P5/, Zuletzt abgerufen am 23. 12. 2009.

[IRC03] Albert Ip, Allyn Radford, and Enrico Canale. Overcoming the presentation mosaic effect of multi-use sharable content objects. In *20 thAnnual Conference of the Australasian Society for Computers in Learning in Tertiary Education*, 2003.

[Jac01] Paul Jaccard. Étude comparative de la distribution florale dans une portion des alpes et des jura. *Bulletin del la Société Vaudoise des Sciences Naturelles*, 37:547–579, 1901.

[Jon72] Karen Spaerck Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28:11–21, 1972.

[KBH+05] David Karger, Karun Bakshi, David Huynh, Dennis Quan, and Vineet Sinha. Haystack: A general purpose information management tool for end users of semistructured data. In *Proceedings of CIDR*, 2005.

[KcCT09] Jong Wook Kim, K. Selçuk Candan, and Junichi Tatemura. Efficient overlap and content reuse detection in blogs and online news articles. In *18th International World Wide Web Conference*, April 2009.

[KFK+06] Claus-Peter Klas, Norbert Fuhr, Sascha Kriewel, Hanne Albrechtsen, Giannis Tsakonas, Sarantos Kapidakis, Christos Papatheodorou, Preben Hansen, Laszlo Kovacs, Andras Micsik, and Elin Jacob. An experimental framework for comparative digital library evaluation: the logging scheme. In *JCDL '06: Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries*, pages 308–309, New York, NY, USA, 2006. ACM Press.

[KFKS05] Claus-Peter Klas, Norbert Fuhr, Sascha Kriewel, and André Schaefer. Daffodil - nutzerorientiertes zugangssystem für heterogene digitale bibliotheken. In *DGI '05*, 2005.

[KH09] Claus-Peter Klas and Matthias Hemmje. Catching the user - user context through live logging in daffodil. In *SIGIR 2009 Workshop: Understanding the User, Boston, USA*, 2009.

[KKH08] Claus-Peter Klas, Sascha Kriewel, and Matthias Hemmje. An experimental system for adaptive services in information retrieval. In *Proceedings of the 2nd International Workshop on Adaptive Information Retrieval (AIR 2008)*, October 2008.

[Kot07] Sotiris B. Kotsiantis. Supervised machine learning: A review of classification techniques. *Informatica*, 31:249–268, 2007.

[Kuh04] Rainer Kuhlen. *5. Auflage des Handbuchs Grundlagen von Information und Dokumentation*, chapter Information, pages 3–20. Saur-Verlag: MÃ¼nchen etc., 2004.

[KVD06] Joris Klerkx, Katrien Verbert, and Erik Duval. Visualizing reuse: More than meets the eye. In *Proceedings of the 6th International Conference on Knowledge Management (I-KNOW) 2006*, pages 489–497, Graz, Austria, September 2006.

[KWIK03] Reinhard Kronsteiner, Edgar Weippl, Ismail Khalil Ibrahim, and Gabriele Kotsis. Can p2p deliver what web repositories promised: Global sharing of e-learning content? In *iiWAS*, 2003.

[LAH+02] Elizabeth D. Liddy, Eileen Allen, Sarah Harwell, Susan Corieri, Ozgur Yilmazel, N. Ercan Ozgencil, Anne Diekema, Nancy McCracken, Joanne Silverstein, and Stuart Sutton. Automatic metadata generation & evaluation. In *SIGIR '02: Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 401–402, New York, NY, USA, 2002. ACM.

[LBPS08] Fabian Loose, Steffen Becker, Martin Potthast, and Benno Stein. Retrieval-Technologien für die Plagiaterkennung in Programmen. In Joachim Baumeister and Martin Atzmüller, editors, *Proceedings of the Information Retrieval Workshop at LWA 2008*, Technical Report 448, pages 5–12. University of Würzburg, Germany, October 2008.

[Lev66] Vladimir I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8):707–710, 1966.

[LFBG07] Robert Lokaiczyk, Andreas Faatz, Arne Beckhaus, and Manuel Goertz. Enhancing just-in-time e-learning through machine learning on desktop context sensors. In Boicho N. Kokinov, Daniel C. Richardson, Thomas Roth-Berghofer, and Laure Vieu, editors, *Sixth International and Interdisciplinary Conference on Modeling and Using Context (CONTEXT)*, volume 4635 of *Lecture Notes in Computer Science*, pages 330–341, Roskilde University, Denmark, 2007. Springer.

[LfC97] Ching-Yung Lin and Shih fu Chang. A robust image authentication method distinguishing jpeg compression from malicious manipulation. *IEEE Transactions on Circuits and Systems of Video Technology*, 11:153–168, 1997.

[LG08] Robert Lokaiczyk and Manuel Goertz. Extending low level context events by data aggregation. In *Proceedings of I-KNOW '08 and I-MEDIA '08*, 2008.

[LGF+07] Robert Lokaiczyk, Eicke Godehardt, Andreas Faatz, Manuel Goertz, Andrea Kienle, Martin Wessner, and Armin Ulbrich. Exploiting context information for identification of relevant experts in collaborative workplace-embedded e-learning environments. In Erik Duval, Ralf Klamma, and Martin Wolpers, editors, *Proceedings of Second European Conference on Technology Enhanced Learning (EC-TEL)*, volume 4753 of *Lecture Notes in Computer Science*, pages 217–231, Crete, Greece, 2007. Springer.

[LHRS07] Lasse Lehmann, Tomas Hildebrandt, Christoph Rensing, and Ralf Steinmetz. Capturing, management and utilization of lifecycle information for learning resources. In E. Duval, R. Klamma, and M. Wolpers, editors, *Creating New Learning Experiences on a Global Scale, Second European Conference on Technology Enhanced Learning, EC-TEL 2007*, pages 187–201, Berlin, Heidelberg, Sep 2007. Springer.

[Lib08] Paul Libbrecht. A model of re-use of e-learning content. In Pierre Dillenbourg and Marcus Specht, editors, *EC-TEL*, volume 5192 of *Lecture Notes in Computer Science*, pages 222–233, Maastricht, Netherlands, 2008. Springer.

[Lie01] Michael Liepert. *Rechte, Benutzerrollen und Inhaltsversionierung für verteilte Multimedia-Autorensysteme*. PhD thesis, Technische Universitaet Darmstadt, Dec 2001.

[LK77] J. Richard Landis and Gary G. Koch. The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174, March 1977.

[LMD01] Caroline Lyon, James Malcolm, and Bob Dickerson. Detecting short passages of similar text in large document collections. In Lillian Lee and Donna Harman, editors, *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*, pages 118–125, Pittsburg, PA USA, 2001.

[Lon02] Phillip D. Long. Opencourseware: Simple idea, profound implications. *Syllabus*, 15(6), 2002.

[LRS07] Lasse Lehmann, Christoph Rensing, and Ralf Steinmetz. Lifecycle information management and utilization in an authoring by aggregation environment. In *Proceedings of World Conference on Educational Multimedia, Hypermedia, EDMEDIA*, 2007.

[LSY03] Greg Linden, Brent Smith, and Jeremy York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80, 2003.

[Mai78] David Maier. The complexity of some problems on subsequences and supersequences. *J. ACM*, 25(2):322–336, 1978.

[Man94] Udi Manber. Finding similar files in a large file system. In *WTEC'94: Proceedings of the USENIX Winter 1994 Technical Conference on USENIX Winter 1994 Technical Conference*, pages 2–2, Berkeley, CA, USA, 1994. USENIX Association.

[MBC+05] Donald Metzler, Yaniv Bernstein, Bruce W. Croft, Alistair Moffat, and Justin Zobel. Similarity measures for tracking information flow. In *CIKM '05: Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 517–524, New York, NY, USA, 2005. ACM.

[McC04] Gord McCalla. The ecological approach to the design of e-learning environments: Purpose-based capture and use of information about learners. *Journal of Interactive Media in Education*, 7:171–193, 2004.

[Mey08] Marek Meyer. *Modularization and Multi-Granularity Reuse of Learning Resources*. PhD thesis, Technische Universität Darmstadt, Sep 2008.

[MHP09] Ronald Maier, Thomas Haedrich, and Rene Peinl. *Enterprise Knowledge Infrastructures - Information and Communication Technologies for Knowledge Work*. Springer Berlin Heidelberg, 2009.

[MHRS06] Marek Meyer, Tomas Hildebrandt, Christoph Rensing, and Ralf Steinmetz. Requirements and an architecture for a multimedia content re-purposing framework. In Wolfgang Nejdl and Klaus Tochtermann (Edits.), editors, *First European Conference on Technology Enhanced Learning - EC-TEL 2006*, pages 500–505, Berlin/Heidelberg, Oct 2006. Springer Verlag.

[MHT+05] Harald Mayer, Werner Haas, Georg Thallinger, Stefanie Lindstaedt, and Klaus Tochtermann. Aposdle - advanced process-oriented self-directed learning environment. In *EWIMT 2005: 2nd European Workshop on the Integration of Knowledge, Semantic and Digital Media Technologies*, London, 2005.

[Mit97] Tom Mitchell. *Machine Learning*. McGraw-Hill Education (ISE Editions), October 1997.

[MKSW99] John Makhoul, Francis Kubala, Richard Schwartz, and Ralph Weischedel. Performance measures for information extraction. In *In Proceedings of DARPA Broadcast News Workshop*, pages 249–252, 1999.

[ML10] Arno Mittelbach and Lasse Lehmann. Shinglecloud library for approximate string matching. Online: http://www.kom.tu-darmstadt.de/en/downloads/software/shinglecloud/, 2010. Zuletzt abgerufen am 13.01.2010.

[MOD07] Michael Meire, Xavier Ochoa, and Erik Duval. Samgi: Automatic metadata generation 2.0. In *Proceedings of World Conference on Educational Multimedia, Hypermedia*, pages 1195–1204, Vancouver, Canada, 2007.

[Mos02] Marie-Luise Moschgath. *Kontextabhaengige Zugriffskontrolle fuer Anwendungen in Ubiquitous Computing*. PhD thesis, KOM - Multimedia Communications Lab, Technische Universitaet Darmstadt, 2002.

[MP05] Ronald Maier and Rene Peinl. Semantic description of documents in enterprise knowledge infrastructures. In Klaus P. Jantke, Klaus-Peter Faehnrich, and Wolfgang S. Wittig, editors, *Leipziger Informatik-Tage*, volume 72 of *LNI*, pages 358–366. GI, 2005.

[MRS06] Marek Meyer, Christoph Rensing, and Ralf Steinmetz. Supporting modularization and aggregation of learning resources in a scorm compliance mode. In X. Kinshuk, R. Koper, P. Kommers, P. Kischner, D.G. Sampson, and W. Didderen (Edits.), editors, *Advanced Learning Technologies, 2006. Sixth International Conference on*, pages 933 – 935, Los Alamitos, Jul 2006. IEEE Computer Society Press.

[MRS07] Marek Meyer, Christoph Rensing, and Ralf Steinmetz. Improving authoring-by-aggregation and using aggregation context for query expansion. In E. Duval, R. Klamma, and M. Wolpers, editors, *Creating New Learning Experiences on a Global Scale, Second European Conference on Technology Enhanced Learning, EC-TEL 2007*, pages 505–510, Berlin, Heidelberg, Sep 2007. Springer.

[MRT07] Josef Makolm, Doris Reisinger, and Klaus Tochtermann. Forschungsprojekt dyonipos: Wissenszentrierte unterstützung von kollaborativen prozessen. In E. Schweighofer, Geist A., and G. Hendl, editors, *Tagungsband zum Internationalen Rechtsinformatik Symposium*, 2007.

[Mue06] Normen Mueller. An ontology-driven management of change. In *In Wissens- und Erfahrungsmanagement, LWA (Lernen, Wissensentdeckung, Adaptivitaet) conference proceedings*, pages 186–193, Hildesheim, Germany, 2006.

[MZ07] Kurt Maly and Mohammad Zubair. Automated metadata extraction for personal publishing in kepler. In Craig Montgomerie and Jane Seale, editors, *Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications 2007*, pages 4500–4504, Vancouver, Canada, June 2007. AACE.

[MZRS07] Marek Meyer, Birgit Zimmermann, Christoph Rensing, and Ralf Steinmetz. An interactive tool for supporting modularization of scorm-based learning resources. In *Proceedings of ED-MEDIA 2007*, pages 3164–3171. AACE, Jun 2007.

[Nav01] Gonzalo Navarro. A guided tour to approximate string matching. *ACM Comput. Surv.*, 33(1):31–88, 2001.

[NHi09] NHibernate. The nhibernate persistence api. Online: https://www.hibernate.org/343.html, Zuletzt abgerufen am 17.11.2009.

[NMD05] Jehad Najjar, Michael Meire, and Erik Duval. Attention metadata management: Tracking the use of learning objects through attention.xml. In *Proceedings of the ED-MEDIA 2005 World Conference on Educational Multimedia, Hypermedia and Telecommunications*, 2005.

[NP05] Wolfgang Nejdl and Raluca Paiu. I know i stored it somewhere - contextual information and ranking on our desktop. In *Proceedings of 8th International Workshop of the EU DELOS Network of Excellence on Future Digital Library Management Systems*, pages 118–124, Schloss Dagstuhl, Germany, 2005.

[NT95] Ikujiro Nonaka and Hirotaka Takeuchi. *The Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation*. Oxford University Press, USA, May 1995.

[NWD06a] Jehad Najjar, Martin Wolpers, and Erik Duval. Attention metadata: Collection and management. In *Proceedings of the WWW2006 workshop on Logging Traces of Web Activity: The Mechanics of Data Collection, Edinburgh, Scotland*, 2006.

[NWD06b] Jehad Najjar, Martin Wolpers, and Erik Duval. Towards effective usage-based learning applications: Track and learn from users experience(s). In *Proceedings of the IEEE ICALT 2006*, pages 1022–1024, Kerkrade, Netherlands, July 2006.

[NWQ+02] Wolfgang Nejdl, Boris Wolf, Changtao Qu, Stefan Decker, Michael Sintek, Ambjoern Naeve, Mikale Nilsson, Matthias Plamer, and Tore Risch. Edutella - a p2p networking infrastructure based on rdf. In *Proceedings of the WWW2002*. w3c, May 2002.

[OAI02] Reference model for an open archival information system (oais) : Recommendation for space data system standards : Ccsds 650.0-b-1, January 2002.

[OB09] Adaora Okoli and Schandl Bernhard. Extraction of contextual metadata from file system interactions. In *Workshop on Exploitation of Usage and Attention Metadata (co-located with Informatik 2009)*, LÃ¼beck, Germany, 9 2009. Gesellschaft fÃ¼r Informatik e.V.

[OD06] Xavier Ochoa and Erik Duval. Use of contextualized attention metadata for ranking and recommending learning objects. In *CAMA '06: Proceedings of the 1st international workshop on Contextualized attention metadata: collecting, managing and exploiting of rich usage information*, pages 9 – 16, Arlington, VA USA, 2006.

[PBMW99] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999. Previous number = SIDL-WP-1999-0120.

[PM08] Elin Ronby Pedersen and David W. McDonald. Relating documents via user activity: the missing link. In *IUI '08: Proceedings of the 13th international conference on Intelligent user interfaces*, pages 389–392, New York, NY, USA, 2008. ACM.

[PMP02] Lutz Prechelt, Guido Malpohl, and Michael Philippsen. Finding plagiarisms among a set of programs with jplag. *Journal of Universal Computer Science*, 8(11):1016–1038, 2002.

[Pol03] Pithamber R. Polsani. Use and abuse of reusable learning objects. *Journal of Digital Information*, 3(4), 02 2003.

[Pro09] SHAMAN Project. Sustaining heritage access through multivalent archiving. Online: http://shaman-ip.eu/shaman/, Zuletzt abgerufen am 21. 12. 2009.

[Qui93] J. Ross Quinlan. *C4.5: Programs for Machine Learning (Morgan Kaufmann Series in Machine Learning)*. Morgan Kaufmann, 1 edition, January 1993.

[RBH+05] Christoph Rensing, Sonja Bergsträßer, Tomas Hildebrandt, Marek Meyer, Birgit Zimmermann, Andreas Faatz, Lasse Lehmann, and Ralf Steinmetz. Re-use and re-authoring of learning resources - definitions and examples. Technical Report KOM-TR-2005-02, TU Darmstadt - Multimedia Communications Lab, Darmstadt, Nov 2005.

[RDL09] Andreas S. Rath, Didier Devaurs, and Stefanie N. Lindstaedt. Detecting real user tasks by training on laboratory contextual attention metadata. In *Exploitation of Usage and Attention Metadata, Informatik '09*, 2009.

[RGRK04] Sean Rhea, Dennis Geels, Timothy Roscoe, and John Kubiatowicz. Handling churn in a dht. In *In Proceedings of the USENIX Annual Technical Conference*, 2004.

[RKL+08] Andreas Rath, Mark Kröll, Stefanie Lindstaedt, Nicolas Weber, Michael Granitzer, and Olivia Dietzel. Context-aware knowledge services. In *Proceedings of Computer Human Interaction (CHI 2008), Workshop on Personal Information Management: PIM 2008*, 2008.

[Roe09] Steven Roebert. C# file browser. Online: http://www.codeproject.com/KB/miscctrl/File-Browser.aspx, Zuletzt abgerufen am 26.11.2009.

[RZM+08] Christoph Rensing, Birgit Zimmermann, Marek Meyer, Lasse Lehmann, and Ralf Steinmetz. Wiederverwendung von multimedialen lernressourcen im re-purposing und authoring by aggregation. In Pavlina Chikova Peter Loos, Volker Zimmermann, editor, *Prozessorientiertes*

Authoring Management: Methoden, Werkzeuge und Anwendungsbeispiele für die Erstellung von Lerninhalten, pages 19–40. Logos Verlag, Berlin, Jan 2008.

[Sau04] Leo Sauermann. The gnowsis semantic desktop for information integration. In *Proceedings of the IOA Workshop of the WM2005 Conference*, 2004.

[SBD05] Leo Sauermann, Ansgar Bernardi, and Andreas Dengel. Overview and outlook on the semantic desktop. In *Proceedings of the 1st Workshop on The Semantic Desktop at the ISWC 2005 Conference*, 2005.

[SC08] Jangwon Seo and W. Bruce Croft. Local text reuse detection. In *Proceedings of SIGIR '08*, Singapore, July 2008. ACM, ACM Press.

[SE05] Ralf Steinmetz and Klaus Wehrle (Edits.). *Peer-to-Peer Systems and Applications*. Springer, Sep 2005.

[Sen09] Ulrich Sendler. *Das PLM-Kompendium - Referenzbuch des Produkt-Lebenszyklus-Managements*. Springer Berlin Heidelberg, 2009.

[SHRS08] Johannes Schmitt, Matthias Hollick, Christoph Roos, and Ralf Steinmetz. Adapting the user context in realtime: Tailoring online machine learning algorithms to ambient computing. *Mobile Networks and Applications*, Volume 13, Number 6 / Dezember 2008, Oct 2008.

[SKR01] J. Ben Schafer, Joseph A. Konstan, and John Riedl. E-commerce recommendation applications. *Data Min. Knowl. Discov.*, 5(1-2):115–153, 2001.

[SR06] Daniel Stutzbach and Reza Rejaie. Understanding churn in peer-to-peer networks. In *Proceedings of the 6th ACM SIGCOMM on Internet measurement*, pages 189–202. ACM Press New York, NY, USA, 2006.

[Ste00] Ralf Steinmetz. *Multimedia-Technologie: Grundlagen, Komponenten und Systeme*. Springer Verlag, Oct 2000. 3. Auflage (erstmalig mit CD).

[Ste02] Achim Steinacker. *Medienbausteine für web-basierte Lernsysteme*. PhD thesis, TU Darmstadt, Jan 2002.

[Str04] Allard Strijker. *Reuse of learning objects in context: human and technical aspects*. PhD thesis, University of Twente, Enschede, September 2004.

[SWA03] Saul Schleimer, Daniel S. Wilkerson, and Alex Aiken. Winnowing: Local algorithms for document fingerprinting. In *Proceedings of SIGMOD 2003*, San Diego,CA, June 2003. ACM, ACM Press.

[Syr01] Apostolos Syropoulos. Mathematics of multisets. In *WMP '00: Proceedings of the Workshop on Multiset Processing*, pages 347–358, London, UK, 2001. Springer-Verlag.

[TND+03] Stefaan Ternier, Filip Neven, Erik Duval, Maciej Macowicz, and Norbert Ebel. Web services for Learning Object Repositories: a Case Study - the ARIADNE Knowledge

Pool System. In *Proceedings of the Twelfth International World Wide Web Conference, WWW2003, Budapest, Hungary, 20-24 May 2003 - Posters*, pages 203–204, 2003. URL: http://www2003.org/cdrom/papers/poster/p203/WWW2003

[TSQ07] Eleftheria Tomadaki, Peter J. Scott, and Kevin Quick. Attention metadata visualizations: Plotting attendance and reuse. In *CAMA 2007: Proceedings of the 2nd international Workshop on Contextualized Attention Metadata: personalized access to digital resources*, pages 27–30, 2007.

[Tur09] Lars Arne Turczyk. *Information Lifecycle Management - Eine Methode zur Wertzuweisung von Dateien*. PhD thesis, TU Darmstadt Elektrotechnik und Informationstechnik: Multimedia Kommunikation, 2009.

[UF98] Lyle H. Ungar and Dean P. Foster. Clustering methods for collaborative filtering. In *Proceedings of the Workshop on Recommendation Systems*. AAAI Press, Menlo Park California, 1998.

[VGJD05] Katrien Verbert, Dragan Gasevic, Jelena Jovanovic, and Erik Duval. Ontology-based learning content repurposing. In *Poster Session at the International World Wide Web Conference Committee*, 2005.

[VH07] Tobias Vogel and Matthias Hemmje. "wissensbasiertes und prozessorientiertes innovationsmanagement" (wpim) - herausforderungen und einsatz in der domäne automotiv. In M. Bentele, R. Hochreiter, G. Riempp, P. Schütt, and M. Weber, editors, *Tagungsband "Mehr Wissen - Mehr Erfolg"*, pages 485–493. CMP-WEKA Verlag & Co. KG, Poing, 2007.

[VJGD05] Katrien Verbert, Jelena Jovanovic, Dragan Gasevic, and Erik Duval. Repurposing learning object components. In *On the Move to Meaningful Internet Systems 2005: OTM Workshops*, 2005.

[VOD08] Katrien Verbert, Xavier Ochoa, and Erik Duval. The alocom framework: Towards scalable content reuse. *Journal of Digital Information*, 9, 2008.

[WA02] Michael Widenius and Davis Axmark. *Mysql Reference Manual*. O'Reilly & Associates, Inc., Sebastopol, CA, USA, 2002.

[Wil02] David A. Wiley. *Connecting learning objects to instructional design theory: a definition, a metaphor, and a taxonomy.* The Agency for Instructional Technology, Bloomington, IN., 2002.

[Wis93] Michael J. Wise. Running karp-rabin matching and greedy string tiling. Technical report, Basser Department of Computer Science - The University of Sydney, 1993.

[WMND06] Martin Wolpers, Gunnar Martin, Jehad Najjar, and Erik Duval. Attention metadata in knowledge and learning management. In *Proceedings of I-Know 2006*, pages 403–410, Graz, Austria, 2006.

[WNVD07] Martin Wolpers, Jehad Najjar, Katrien Verbert, and Erik Duval. Tracking actual usage: the attention metadata approach. *International Journal Educational Technology and Society*, 11:106–121, 2007.

[ZBC+05] Volker Zimmermann, Kathrin Bergenthal, Pavlina Chikova, Didier Hinz, Lasse Lehmann, Katrina Leyking, Gunnar Martin, and Christoph Rensing. Authoring management platform explain. a new learning technology approach for efficient content production integrating authoring tools through a web-based process and service platform. In *Proceedings of the ARIADNE PROLEARN WORKSHOP*, 2005.

[Zim08] Birgit Zimmermann. *Pattern-basierte Prozessbeschreibung und -unterstützung: Ein Werkzeug zur Unterstützung von Prozessen zur Anpassung von E-Learning-Materialien*. PhD thesis, Technische Universität Darmstadt, Dec 2008.

[ZRS06] Birgit Zimmermann, Christoph Rensing, and Ralf Steinmetz. Format-uebergreifende anpassungen von elektronischen lerninhalten. In *Proceedings of the DeLFI 2006 - die 4. e-Learning Fachtagung Informatik*, 2006.

[ZRS07] Birgit Zimmermann, Christoph Rensing, and Ralf Steinmetz. Ein werkzeug zur unterstützung der anpassung existierender e-learning materialien. In Christian Eibl, Johannes Magenheim, Sigrid Schubert, and Martin Wessner, editors, *DeLFI 2007: 5. e-Learning Fachtagung Informatik*, pages 293–294, Köllen, Bonn, Sep 2007. GI, Lecture Notes in Informatics (LNI).

Abkürzungsverzeichnis

ALOCoM	Abstract Learning Object Content Model
API	Application Programming Interface / Programmierschnittstelle
APOSDLE	Advanced Process-Oriented Self-Directed Learning Environment
CAM	Contextualised Attention Metadata
CVS	Concurrent Versions System
DAX	Deutscher Aktien IndeX
DC	Dublin Core
DCT	Discrete Consine Transform - Diskrete Kosinustransformation
DMS	Document Management System
DYONIPOS	DYnamic ONtology based Integrated Process OptimiSation
GST	Greedy String Tiling
GUID	Globally Unique Identifier
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
hylOs	hypermedia learning Object system
ID	Identifier / Identifikator
IEEE	Institute of Electrical and Electronics Engineers, Inc.
IP	Internet Protocol
LCMS	Learning Content Management System / Lern-Content-Management-System
LCS	Longest Common Subsequence
LIS	Lifecycle Information System
LMS	Learning Management System
LO	Learning Object / Lernobjekt
LOM	Learning Object Metadata

LOR	Learning Object Repository / Lernobjekt-Repository
LR	Learning Resource / Lernressource
MAC	Media Access Control
METER	MEasuring Text REuse
MML	Minimal Match Length / Minimallänge
MTL	Minimal Tile Length
ND	Non Derived
OAIS	Open Archival Information System
P2P	Peer-to-Peer
PC	Personal Computer
PD	Partially Derived
PDF	Portable Document Format
PIM	Personal Information Management / Persönliches Informationsmanagement
SAmgI	Simple Automatic Metadata Generation Interface
SC	ShingleCloud
SCORM	Sharable Content Object Reference Model
SHAMAN	Sustaining Heritage Access through Multivalent ArchiviNg
SQL	Structured Query Language
SVN	Subversion
TeNDaX	Text Native Database eXtension
TF-IDF	Term Frequency - Inverse Document Frequency
TP	Toleranzparameter
TREC	Text REtrieval Conference
URL	Uniform Resource Locator
WD	Wholly Derived
WPIM	Wissensbasiertes Prozessorientiertes Innovationsmanagement
WWW	World Wide Web
XML	Extensible Markup Language

Teil V.

Appendix

Tabelle A.1.: String-Matching-Ansätze auf dem METER-Korpus (einfache Vorverarbeitung)

Ansatz	Wholly-Derived	Partially-Derived	Non-Derived	Gewichtetes Gesamt-F1-Maß	Konfidenz-intervall
1-Gram	0,7040	0,6386	0,6336	**0,6598**	± 0,0010
2-Gram	0,6203	0,5625	0,5735	**0,5846**	± 0,0014
3-Gram	0,6255	0,6111	0,5540	**0,6037**	± 0,0010
GST (Mindestlänge = 1)	0,5970	0,6396	0,6217	**0,6212**	± 0,0012
GST (Mindestlänge = 2)	0,5993	0,6185	0,6098	**0,6100**	± 0,0006
GST (Mindestlänge = 3)	0,6788	0,6057	0,5466	**0,6179**	± 0,0010
GST (Mindestlänge = 4)	0,6539	0,6165	0,5191	**0,6082**	± 0,0012
ShingleCloud 2,3,4	0,6161	0,6136	0,6096	**0,6136**	± 0,0012
ShingleCloud 1,1,2	0,6956	0,5890	0,5240	**0,6112**	± 0,0026
ShingleCloud 1,2,0	0,6026	0,6503	0,6239	**0,6284**	± 0,0009
ShingleCloud 3,1,3	0,6814	0,6177	0,4818	**0,6101**	± 0,0013

A Weitere Daten der Evaluation von Validierungsalgorithmen

Im Folgenden finden sich weitere Daten zur Auswertung der Validierungsalgorithmen MiLe und ShingleCloud.

A.1 Weitere Auswertung von ShingleCloud auf dem METER-Korpus

Es wurden zusätzliche Auswertungen des ShingleCloud-Ansatzes auf dem Meter-Korpus mit verändertem Preprocessing durchgeführt. Wenn keine Stoppwortliste verwendet wird, sondern lediglich einfache Vorverarbeitung (Säubern von nicht alphanumerischen Tokens, Entfernung von Satzzeichen) angewandt wird, ergeben sich Ergebnisse wie in Tabelle A.1 gezeigt. Insbesondere 1-Gram bietet mit dieser Art der Vorverarbeitung im Vergleich deutlich bessere Qualität als die übrigen Ansätze. Das hängt vor allem mit den beschriebenen Eigenschaften des Korpus zusammen (siehe 5.5.2). Der GST-Algorithmus liefert auf dem METER-Korpus mit dieser Art der Vorverarbeitung gute Ergebnisse, auch für Minimallängen größer als zwei. Auch ShingleCloud zeigt hier allgemein gute Ergebnisse, wenn eine kleine Fenstergröße gewählt wird. Im Allgemeinen variieren die Ergebnisse der Algorithmen auf dem METER-Korpus für verschiedene Arten der Vorverarbeitung vergleichsweise stark.

Tabelle A.2.: Vergleich der Fingerprinting-Ansätze auf dem METER-Korpus mit einfacher Vorverarbeitung

Ansatz	WD	PD	ND	Gewichtetes Mittel	Konfidenz-intervall
1-gram	0,704	0,639	0,634	**0,6598**	± 0,0004
2-gram	0,620	0,562	0,574	**0,5846**	± 0,0009
3-gram	0,625	0,611	0,554	**0,6037**	± 0,0005
0 mod 4 (k = 3)	0,645	0,605	0,350	**0,5638**	± 0,0008
0 mod 4 (k = 2)	0,624	0,538	0,428	**0,5437**	± 0,0004
0 mod 6 (k = 3)	0,657	0,569	0,481	**0,5800**	± 0,0007
0 mod 6 (k = 2)	0,650	0,564	0,457	**0,5702**	± 0,0011
MiLe4,5,1 (4 bits, mml 5, TP 1)	0,628	0,601	0,473	**0,5822**	± 0,0007
MiLe5,4,0 (5 bits, mml 4)	0,630	0,617	0,544	**0,6058**	± 0,0007
MiLe6,3,0 (6 bits, mml 3)	0,610	0,535	0,552	**0,5640**	± 0,0006
Baseline	0,3402	0,4442	0,2155	**0,4442**	-

A.2 Weitere Auswertung von MiLe auf dem METER-Korpus

Wenn die bei der Vorverarbeitung der Dokumente bei der Auswertung des MiLe-Fingerprints auf dem METER-Korpus die Stoppwortfilterung weggelassen wird, ändern sich die Ergebnisse etwas. Es ist jedoch auch in diesem Fall möglich MiLe so zu parameterisieren, dass der Algorithmus das zweitbeste der Ergebnisse liefert. Interessant ist, dass MiLe hier auch mit einer Bitlänge von vier eine vergleichsweise gute Qualität aufweist. Dass dies hier funktioniert und bei einer aktivierten Filterung von Stoppwörtern nicht, hat folgenden Grund: Durch die Filterung von Stoppwörtern werden oftmals übereinstimmende Sequenzen verkürzt, so dass sie in einigen Fällen nicht mehr die erforderliche Minimallänge erreichen. Dadurch werden diese übersehen, wenn die Minimallänge entsprechend ist. Bei geringeren Bitlängen benötigt der MiLe-Algorithmus zum Ausgleich jedoch eine entsprechend hohe Minimallänge. Diese Konfiguration schneidet dann auf einem Korpus, dessen Stoppwörter gefiltert wurden, schwächer ab. Demgegenüber schneidet eine Konfiguration mit größerer Bitlänge und entsprechend kleiner Minimallänge, die auf einem Korpus mit Stoppwortfilterung gute Ergebnisse liefert, auf diesem Korpus auf Grund von Kollisionen schwächer ab.

A.3 Weitere Auswertungen von MiLe auf dem annotierten TREC-Korpus

Das folgende Diagramm zeigt die Auswirkung der Anwendung des Auslassfaktors auf verschiedene Konfigurationen des MiLe-Fingerprints auf dem annotierten TREC-Korpus. Es ist gut zu sehen, wie mit steigendem Auslass-Faktor die Qualität in allen Konfigurationen abnimmt. Die statistischen Schwankungen werden offensichtlich mit zunehmender Bitlänge geringer.

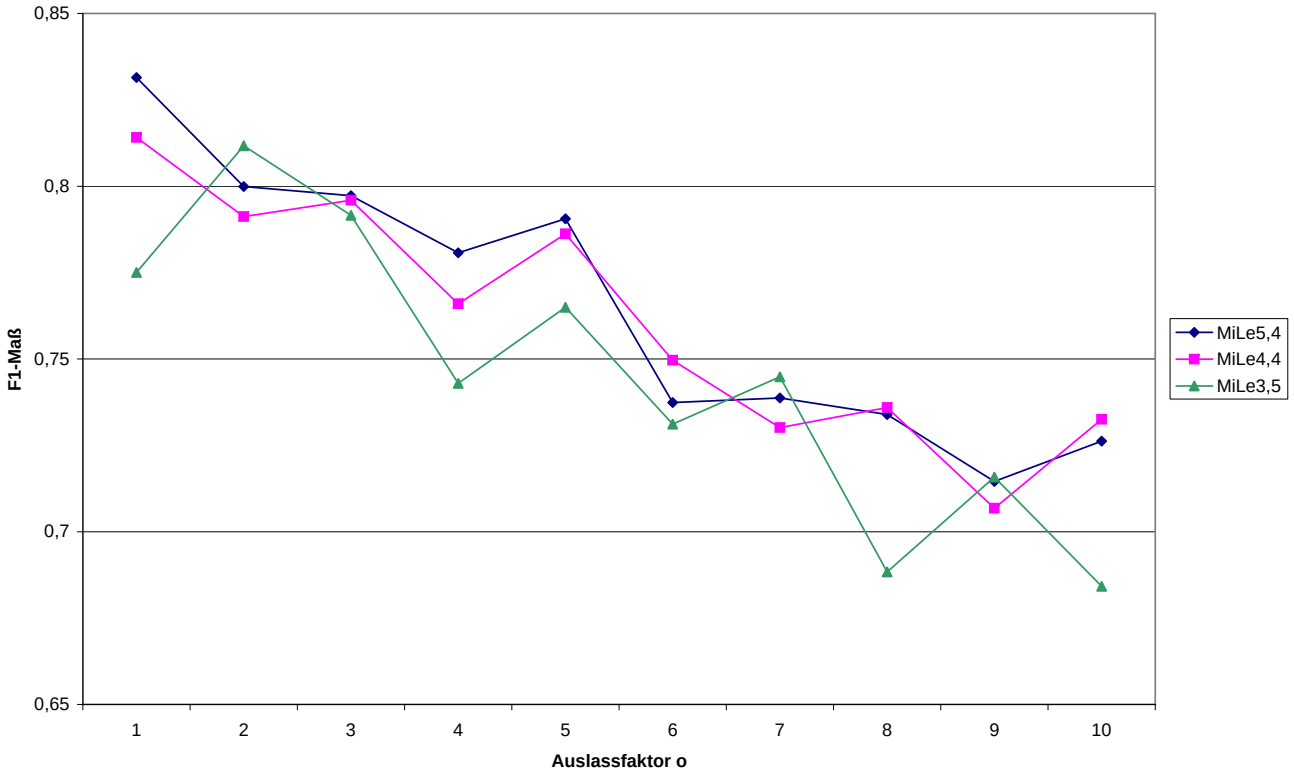

Abbildung A.1.: Auswirkung des Auslassfaktors auf die Qualität von MiLe

Im Vergleich der verschiedenen Ansätze auf dem TREC-Korpus wurden eine Berechnung der Ähnlichkeit auf Multimengen-Basis für K-Gram, sowie der Toleranzparameter für MiLe der Übersichtlichkeit halber weggelassen. Es zeigt sich, dass sowohl 2-Gram als auch 3-Gram von der Multimengen-Berechnung profitieren, wie auch MiLe5,4 und MiLe4,4 vom Einsatz des Toleranzparameters. Abbildung A.2 verdeutlicht dies. An der prinzipeillen Reihenfolge der verglichenen Ansätze und der Signifikanz der Ergebnisse ändert sich jedoch nichts. Auch in diesem Fall ist MiLe signifikant besser als K-Gram.

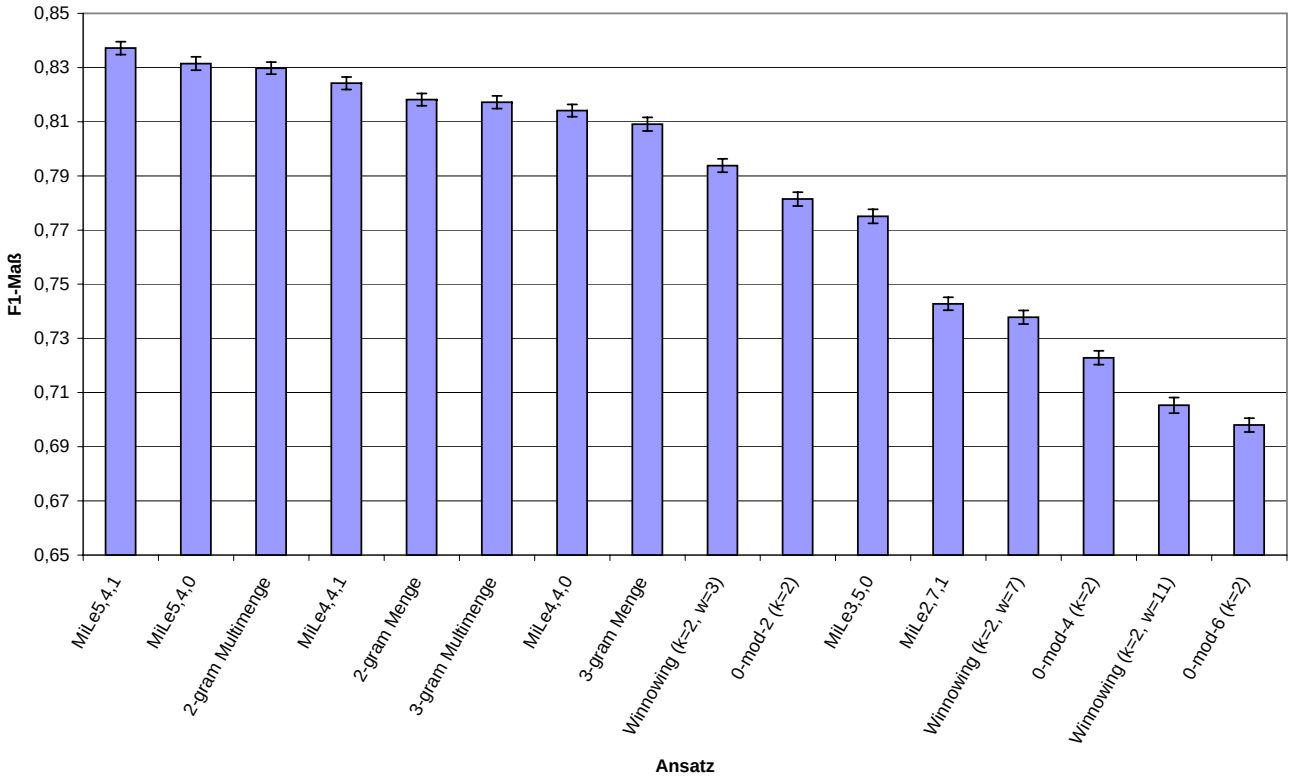

Abbildung A.2.: Vergleich unterschiedlicher Konfigurationen von K-Gram, Winnowing, 0-mod-p und MiLe auf dem annotierten TREC-Korpus

B Liste der eigenen Publikationen

B.1 Journals und Buchkapitel

1. Lasse Lehmann, Arno Mittelbach, Christoph Rensing and Ralf Steinmetz (2010): Capture of Lifecycle Information in Office Applications, Int. J. Technology Enhanced Learning, Vol. 2, Nos. 1/2, pp.41-57.

2. Lasse Lehmann, Tomas Hildebrandt, Christoph Rensing, Ralf Steinmetz: Capture, Management and Utilization of Lifecycle Information for Learning Resources. In: IEEE Transactions on Learning Technologies, vol. 1, no. 1, p. 75-87, March 2008. ISSN 1939-1382.

3. Lasse Lehmann, Christoph Rensing, Ralf Steinmetz: Effektive Nutzung von Medienressourcen aus dem betriebswirtschaftlichen Lebenszyklus eines Produkts zur Modellierung und Produktion von Trainingsinhalten am Beispiel der EXPLAIN-Plattform. In: Peter Loos, Volker Zimmermann, Pavlina Chikova: Prozessorientiertes Authoring Management: Methoden, Werkzeuge und Anwendungsbeispiele für die Erstellung von Lerninhalten, p. 183-200, Logos Verlag, January 2008. ISBN 978-3-8325-1939-1.

4. Christoph Rensing, Birgit Zimmermann, Marek Meyer, Lasse Lehmann, Ralf Steinmetz: Wiederverwendung von multimedialen Lernressourcen im Re-Purposing und Authoring by Aggregation. In: Peter Loos, Volker Zimmermann, Pavlina Chikova: Prozessorientiertes Authoring Management: Methoden, Werkzeuge und Anwendungsbeispiele für die Erstellung von Lerninhalten, p. 19–40, Logos Verlag, January 2008. ISBN 978-3-8325-1939-1.

B.2 Konferenzen und Workshops

5. Lasse Lehmann, Christoph Rensing, Ralf Steinmetz: Capture of Lifecycle Information to Support Personal Information Management. In: Pierre Dillenbourg, Marcus Specht: Times of Convergence: Technologies Across Learning Contexts, Third European Conference on Technology Enhanced Learning, EC-TEL 2008, p. 216-221, Springer, September 2008. ISBN 978-3-540-876-04-5.

6. Lasse Lehmann, Christoph Rensing, Ralf Steinmetz: Capture of Lifecycle Information in Proprietary Office Applications. In: David Massart, Jean-Noël Colin and Frans Van Assche: Proceedings of the Second Second International Workshop on Search and Exchange of e-le@rning Materials (SE@M'08), CEUR-WS, September 2008. ISBN (ISSN) 1613-0073.

7. Lasse Lehmann, Helge Fredrich, Christoph Rensing, Volker Zimmermann, Ralf Steinmetz: Das Authoring Management System EXPLAIN zur ganzheitlichen Unterstützung des Erstellungsprozesses von Trainingsmedien und WBT. In: Christian Eibl, Johannes Magenheim, Sigrid Schubert, Martin

Wessner: DeLFI 2007: 5. e-Learning Fachtagung Informatik, no. P-111, p. 139–150, Lecture Notes in Informatics (LNI), September 2007. ISBN 978-3-88579-205-5.

8. Lasse Lehmann, Tomas Hildebrandt, Christoph Rensing, Ralf Steinmetz: Capturing, Management and Utilization of Lifecycle Information for Learning Resources. In: E. Duval, R. Klamma, and M. Wolpers: Creating New Learning Experiences on a Global Scale, Second European Conference on Technology Enhanced Learning, EC-TEL 2007, p. 187–201, Springer, September 2007. ISBN 978-3-540-75194-6.

9. Lasse Lehmann, Tomas Hildebrandt, Christoph Rensing, Ralf Steinmetz: Utilizing Lifecycle Information for Knowledge Document Management and Retrieval. In: Klaus Tochtermann, Hermann Maurer: Proceedings of the 7th International Conference on Knowledge Management, p. 192-199, J. UCS, September 2007. ISBN 0948-695x.

10. Lasse Lehmann: Lifecycle Information Management for Learning Resources and Knowledge Documents. In: Katherine Maillet and Ralf Klamma: 2nd PROLEARN Doctoral Consortium in Technology Enhanced Learning, p. 28-32, CEUR Workshop Proceedings, September 2007. ISBN 1613-0073. http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-288/p05.pdf.

11. Lasse Lehmann, Christoph Rensing, Ralf Steinmetz: Lifecycle Information Management and Utilization in an Authoring by Aggregation Environment. In: Craig Montgomerie, Jane Seale: Proceedings of World Conference on Educational Multimedia, Hypermedia 2007, p. 3142-3150, June 2007. ISBN 1-880094-62-2.

12. Lasse Lehmann, Abdelhak Aqqal, Christoph Rensing, Ralf Steinmetz: A Content Modelling Language as Basis for the Support of the Overall Content Creation Process. In: Kinshuk X.; Koper, R.; Kommers, P.; Kischner, P.; Sampson, D.G.; Didderen, W. (Edits.): Advanced Learning Technologies, 2006. Sixth International Conference on, no. ISBN 0-7695-2632-2, p. 10-12, IEEE Computer Society Press, July 2006.

13. Sonja Bergsträßer, Tomas Hildebrandt, Lasse Lehmann, Christoph Rensing, Ralf Steinmetz: Virtual Context Based Services for Support of Interaction in Virtual Worlds. In: Grenville Armitage: Netgames 2007, 6th Annual Workshop on Network and Systems Support for Games, Melbourne, Australia, p. 111–116, September 2007. ISBN 978-0-9804460-0-5.

14. Pavlina Chikova, Katrina Leyking, Peter Loos, Eva-Maria Bruch, Lasse Lehmann: Reengineering der Content-Erstellungsprozesse in Industrieunternehmen durch Content-Modellierung: Fallbeispiel. In: 8. Internationale Tagung Wirtschaftsinformatik WI 2007; Prozess-Engineering (eLearning - Anforderungen an das Content Management), February 2007.

15. Katrina Leyking, Pavlina Chikova, Lasse Lehmann: Technology-Enhanced Learning meets CIM - Integrated Content Development Processes. In: Workshop-Proceedings der DeLFI2006, September 2006.

16. Christoph Rensing, Stephan Tittel, Lasse Lehmann, Ralf Steinmetz: Ein System zur Realisierung expliziten Lerner-Autor Feedbacks im E-Learning. In: Christoph Rensing: Proceedings der Pre-

Conference Workshops der 4. e-Learning Fachtagung Informatik DeLFI 2006, p. 27–34, Logos, September 2006. ISBN ISBN 3-8325-1330-2.

17. Volker Zimmermann, Kathrin Bergenthal, Pavlina Chikova, Didier Hinz, Lasse Lehmann, Katrina Leyking, Gunnar Martin, Christoph Rensing: Authoring Management Platform EXPLAIN. A new learning technology approach for efficient content production integrating authoring tools through a web-based process and service platform. In: Proceedings of the ARIADNE PROLEARN WORKSHOP, December 2005.

B.3 Technical Reports

18. Christoph Rensing, Sonja Bergsträßer, Tomas Hildebrandt, Marek Meyer, Birgit Zimmermann, Andreas Faatz, Lasse Lehmann, Ralf Steinmetz: Re-Use and Re-Authoring of Learning Resources - Definitions and Examples. no. KOM-TR-2005-02, November 2005.

C Lebenslauf des Verfassers

Persönliche Daten

Name:	Lasse Matthias Lehmann
Geburtsdatum:	28. Mai 1979
Geburtsort:	VS-Villingen
Nationalität:	Deutsch
Familienstand:	verheiratet, 2 Kinder

Akademischer Werdegang

seit 07/2005	Technische Universität Darmstadt, Fachgebiet Multimedia Kommunikation - Wissenschaftlicher Mitarbeiter
1999 – 2005	Technische Universität Darmstadt Studiengang: Elektrotechnik (Diplom) Abschluss: Diplom-Ingenieur
1989 – 1998	Gymnasium am Hoptbühl VS-Villingen, Abschluss: Abitur

Berufliche Tätigkeiten

seit 07/2005	Technische Universität Darmstadt, Fachgebiet Multimedia Kommunikation - Wissenschaftlicher Mitarbeiter
2004 – 2005	httc e.V. - Wissenschaftliche Hilfskraft
2004	httc e.V. - Fachpraktikum
1999	Mannesmann VDO - Praktikum
1998-1999	DRK Donaueschingen - Zivildienst als Rettungshelfer

D Erklärung laut §9 PromO

Ich versichere hiermit, dass ich die vorliegende Dissertation allein und nur unter Verwendung der angegebenen Literatur verfasst habe. Die Arbeit hat bisher noch nicht zu Prüfungszwecken gedient.

Darmstadt, den 22. Januar 2010 Lasse Lehmann